Process Based Modeller For Axisymmetric Parts
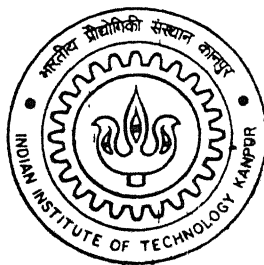
By

Jeevan Vijay Patwa

DEPARTMENT OF MECHANICAL ENGINEERING

Indian Institute of Technology Kanpur

FEBRUARY, 2002

Process Based Modeller For Axisymmetric Parts

A thesis submitted
in partial fulfillment of the requirements
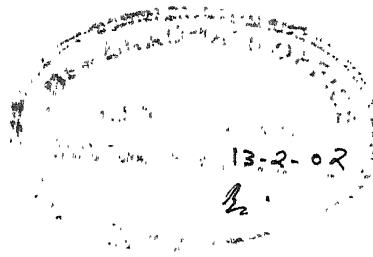for the degree of

Master of Technology

by

Jeevan Vijay Patwa

Department of Mechanical Engineering
Indian Institute of Technology
Kanpur – 208016, India

Feb, 2002

CERTIFICATE

It is certified that the work contained in the thesis entitled, "**Process Based Modeller For Axisymmetric Parts**", by *Mr. Jeevan Vijay Patwa (Roll No. Y010523)* has been carried out under my supervision and that this work has not been submitted else where for a degree.

Dr. S. G. Dhande
Professor,
Department of Mechanical Engineering,
Indian Institute of Technology, Kanpur.

Feb, 2002.

Dedicated

To

My Parents

ACKNOWLEDGEMENT

Abstract

For CAD/CAM integration purpose much work has been done in Feature Recognition and Feature Mapping. After realizing the drawbacks of transfer of data, separate feature recognition algorithms, scholars tried to develop the modelling environment using features as the only basis entities so that it can be directly linked to CAPP system. But for a common user the terminology of features, Boolean operations is bit abstruse Hence an attempt has been made to develop the modeling environment, which uses basic manufacturing processes only as basis entities.

In the present work a process based modeller has been developed for axisymmetric parts. It provides a modelling environment to the user where he can do the modelling in terms of the processes and specified process parameters. Since it considers only axisymmetric parts, the processes considered in the present work are facing, turning, parting, taper turning, boring, drilling and slotting

The objective of the work is to develop a modelling environment, which can be easily integrated with downstream applications like Computer Aided Process Planning. It proposes a new approach towards CAD/CAM integration by using the manufacturing processes as the basis entities for modelling purpose. Since the manufacturing processes are used for building up the model, the application can be easily extended to Computer Aided Process Planning.

The software provides different modelling features like Undo, History List Editing different I/O operations like file save, file open, export, different viewing controls and easy-to-use Graphical User Interface (GUI).

The entire software is designed on ANSI C compiler compatible with both UNIX and WINDOWS environment having an in-built OpenGL support. For modelling purpose OpenGL and GLUT libraries are used whereas GLUI library has been used for developing the Graphical User Interface (GUI).

Table of Contents

.

.

List of Figures

Chapter 1

Introduction and Literature Survey

Smooth and effective integration of computer-aided design (CAD) and computer aided manufacturing (CAM) systems is vital for companies striving for survival in the present increasingly competitive marketplace. Features are generally regarded as a key component in this integration effort. In the design realm, features provide a means for capturing, explicitly, the engineering attributes and relationships between product definition entities. This richer product database is an essential requirement for automating analyses and various design tasks. In the manufacturing realm, features can be linked to manufacturing knowledge of various types. Hence it is possible to automate manufacturing process planning and to generate the detailed operating instructions required by modern productive systems such as CNC machines, flexible manufacturing systems, robots, and CMM inspection equipment.

There are many published definitions of the concept of a feature. Even though these definitions seem to be dissimilar, they all consider features as entities, which are of semantically higher level than the pure geometric elements typically used in solid modelling systems. Geometric elements are solid primitives in constructive solid geometry (CSG) type solid models (blocks, cylinders, spheres, tori) or boundary elements used in boundary representation (B-Rep) type solid models (faces, edges, vertices). Almost universally, the concept of generic feature classes is used, and models are built from instance of generic features. The generic types may be organized into feature taxonomy, often realized as a collection of classes with inheritance of information according to the principles of object oriented programming.

In the type instance approach, feature instances are represented in terms of various feature attributes. Common attributes include the intrinsic geometric attributes of the shape corresponding to the feature (length, width, depth, radius etc.), the position and orientation of the feature with respect to some global coordinate frame, geometric

tolerances, material properties, and references to adjacent and other features. Types contain information shared by all instances of the type. In object-oriented approach to features, this information often is in the form of procedures for computing interesting properties of the instances, such as volume and cost.

A useful categorization of the various definitions is the separation of "top-down" definitions and "bottom-up" definitions. Top-down definitions emphasize the designer's view of features as elementary entities of a part definition, which can be computed on the basis of the feature definition. Bottom-up definitions emphasize features as abstraction of recurring combinations of geometric elements.

The above categorization typically reflects the primary feature creation model used by feature-based modelling systems. Thus top-down definitions correspond with systems utilizing primarily design-by-features method, where models are originally defined in terms of their constituents features. Similarly, bottom-up definitions correspond with systems utilizing feature recognition, where features are extracted from previously generated geometric models.

Manufacturing features

Features can be defined from different viewpoints, such as design, analysis, assembly and function. Because of this, there may be several co-existing feature models of the same product design. This gives rise to the problem of feature mapping, i.e. conversion among the various viewpoints. A manufacturing feature is typically defined as a related geometric element which corresponds to a particular manufacturing method or process, or which can be used to reason about the suitable manufacturing methods or processes for creating that geometry.

The link between manufacturing features and manufacturing knowledge is typically realized through manufacturing process models. For machining, the process models can be organized into a taxonomy containing elementary processes such as milling, drilling, facing and turning. Process models are expressed in terms of the manufacturing resources which can be used to realize the process (machines, tools, fixtures, auxiliary materials), process parameters related to the use of resource (for machining, feed and speed), and

attribute information which can be used to guide the choice of a particular process (such as time and cost) often procedural knowledge is included. such as procedures for computing the process parameters on the basis of feature attribute information An important aspect of process model is representation of the tool kinematics, such as access direction and possible technological constraints.

To implement the link, manufacturing feature types refer to a collection of possible process models, which can be used to generate instances of the feature type. Thus for instance, a hole feature can be linked with process models related to drilling and milling processes.

There are two ways by which product data is prepared for feature based manufacturing applications.

- Recognition of manufacturing features from a solid model
- Mapping a design feature model to manufacturing features.

Considerable research has been done in feature recognition over the last two decades. On the other hand, feature mapping (also known as feature conversion, transformation, transmutation) came about recently, only after design by feature approach became popular. In feature recognition, the solid model is created first, and manufacturing feature recognized, either interactively by the process planer, or automatically. Some hybrid methods have also been advised to combine interactive and automatic recognition. In feature mapping one has the benefit, at least, in theory, of extracting manufacturing information from a richer database. Solid models are only capable of providing nominal geometry, which is all that is available in feature recognition.

So up till now research has been made to first exporting the 3D solid model data from the Modelling software in the neutral file format such as IGES or STEP then incorporating some feature recognition system to extract the features. Once the features are extracted then comes the feature mapping which transforms the features from design domain to manufacturing domain by representing each feature as instance of a particular process and corresponding process parameters. This data is then used for computer aided process

planning for doing the process planning tasks such as sequencing, checking for collision, optimization of process parameters to reduce the time and cost etc.

Some researchers have tried to build feature based modelling systems where they provide the user with the feature library and the Boolean operators with which one can create the model. Since each feature can be regarded as an instance of a particular process, the current software provides the process based modelling in advance to feature based modelling. For a common user the terms like features and Boolean operations may be abstruse but if he has a fair knowledge about the basic manufacturing processes, he can easily create the model using the present software. This software is designed only for the rotational parts.

The processes provided by the software and their basic description is as follows.

Turning

Turning is the process of machining external cylindrical and conical surfaces. The work piece is rotated in a longitudinal fed, single point cutting tool. When tool is fed parallel to the axis of rotation then it is called step turning. It produces cylindrical surface.
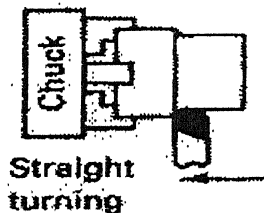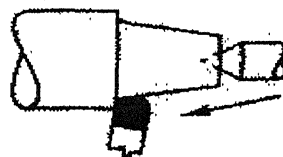
Fig. 1.1 : Step Turning   **Fig. 1.2 : Taper Turning**

Taper Turning

When tool is fed at an angle to the axis of rotation then it is called as taper turning. It produces conical surfaces.

Facing

If the tool is fed at 90° to the axis of rotation using a tool that is wider than the width of the cut the operation is facing and a flat surface is produced.
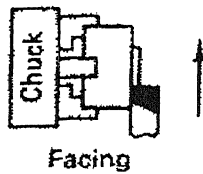
Fig. 1.3 : Facing
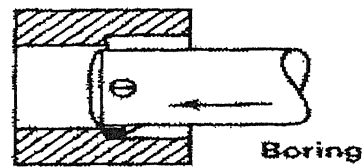
Fig. 1.4 : Boring

Boring

It is a variation of turning. Essentially it is an internal turning. It always involves enlarging an existing hole or removing the eccentricity of hole.

Parting

It is an operation by which one section of work piece is severed from the remainder by means of a cut-off tool. Tool is kept at height of the axis of rotation and fed perpendicular to the axis of rotation.

Slotting / Necking

It is done to produce a neck or slot at an intermediate position of the smaller width, which can't be machined by turning.
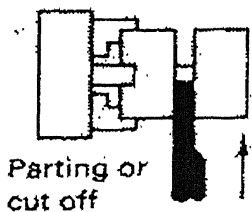
Fig. 1.5 : Parting
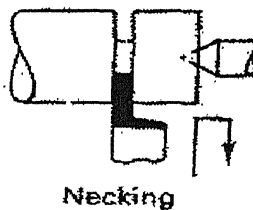
Fig. 1.6 : Necking
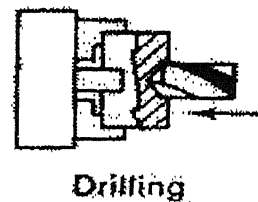
1.7 : Drilling

Drilling

It is done with drill mounted on tailstock and fed against the rotating work piece.

Once the model is created it can be exported to other CAD software or can be exported in 2D profile form, which can be further used for sequencing and process planning operations and generating NC code.

1.1 Literature Survey

A large variety of techniques have been developed for feature recognition. They can be classified as follows

Sectioning methods
Convex hull decomposition
Boundary based methods
Cellular decomposition

1.1.1 Sectioning Methods

This has been done by Murray et al. [1993]. These are applied typically to generate tool paths for 2 ½ D milling. Before this method can be applied, one needs to determine (automatically or manually) that the part is manufacturable by 2 ½ D milling. The path is oriented such that its principle feature directions coincide with the three milling axes. The part volume is sliced with planes parallel to X-Y at fixed ΔZ values, representing a series of tool positions. This results in one or more intersection profiles, representing the part's boundaries. These profiles are classified as "material" or "void" and offset curves generated to form the basis for NC tool path generation.

1.1.2 Convex Hull Decomposition

Originally developed by Woo [1991] this algorithm decomposes a volume by subtracting it from its convex hull and repeating the process for all the resulting volumes. The decomposition for each sub-volume terminates when a null object results. Thus a volume is decomposed into an alternating sum of volumes (ASV). The original algorithm has the problem of non-convergence in many cases, and often resulted in volumes that did not bear resemblance to common manufacturing features. Kim [1992] extended the ASV method by introducing partitioning of non-extreme faces in order to solve the problem of non-convergence. A convex decomposition called Alternating Sum of Volumes with Partitioning (ASVP) is a hierarchical volumetric representation, which is obtained from the boundary information of the given solid by exploiting convexity. Kim and Wilde

[1992] proposed a novel method to recognize volumetric form features intrinsic to the shape of a given solid using ASVP decomposition. Form feature decomposition is converted into corresponding negative components to represent the removal volume for machining applications. The positive to negative conversion is done in top-down manner by abstracting the positive components using half spaces determined by the original faces and combining with the parent negative component. This method was limited to polyhedral solids only due to difficulty of convex hull construction for curved solids. The form feature decomposition serves as a central feature representation, which will form the core of the integration of design and manufacturing. This allows extraction of many context specific feature descriptions to support various product-engineering applications.

1.1.3 Boundary Based Methods

All methods that operate primarily on B-Rep models and use geometric and topological relations between boundary entities have been lumped under this category. For each feature, the geometric and topological conditions that need to be satisfied are identified. To find features in a solid model, the database is searched to see if the conditions corresponding to each feature is present. Sometimes it is more convenient to build a separate data structure, such as face adjacency graph to facilitate the search. Another key concept in feature recognition is entity classification, which specifies the geometric relation between two entities. For example, if two faces meet at an angle under 180°, the edge at which they meet is classified as convex.

The actual mechanics of representing and matching geometric and topological relations may vary considerably. Rule Bases Procedures, graph grammar, syntactical methods, algebraic representations and neural nets have all been employed in specific implementation.

1.1.3.1 Rule Based Approach

The rule based approach (Kyprianou [1980]), (Henderson and Anderson [1984]), (Hwang [1988]), (Vanderbrande [1990]) uses artificial intelligence techniques to develop a set of feature rules. Determining the necessary and sufficient conditions for the feature of

choice and expressing them in a logic statement write rules for feature. Rule based algorithms identify a feature based on certain prespecified rules that are characteristic to feature. The allowed entities expressed in the rule are typically the boundary elements: faces, edges and vertices. Constraint operators typically include parallel, perpendicular, adjacent, equal, concave, convex and other relationships or conditions that these entities must satisfy to define a particular feature. Rule based feature recognition can be used to identify features given that the rule can be written, the rule is correct, a solid model must exist for the part and a search technique is available to match the solid model on the rule conditions.

Choi [1984] used a pattern matching technique in his feature recognition research. He defined a 3D boundary file data structure for representing the solid model. He also defined "start surfaces ", "element surfaces" and "bottom surfaces" for each particular set of "machined surfaces" (manufacturing feature). The 3D boundary file data structure Choi used is essentially a graph representation of a 3D solid model. This approach implied the concept of graph searching. The pre-defined feature definitions are piece-wise, that is there is no geometry constraint specified between start surfaces, element surfaces ad bottom surfaces. A cylindrical protrusion may be mis-recognized as a hole. Also the recognition algorithm is highly feature dependent, which restricts a system developer to design different algorithms for different features.

Henderson [1984] build an expert system for feature recognition in Prolog. He converted B-Rep models and feature definitions into facts and rules of the expert system. The reasoning mechanism of Prolog searches through the facts to recognize features according to the rules. His algorithm first subtracted the part solid model from the original stock. The volume produce by subtraction essentially consists of manufacturing features. The found features are removed from the volume to prevent further recognition. A feature graph is built to keep the removed features in a correct machining order. He also introduced the "entrance face" concept in his work for machining accessibility of features.

Disadvantages of rule based approach are: 1) rules are non-unique to a feature, 2) rules can't be devised for every conceivable feature and 3) recognition involves repeated exhaustive searches of the solid model which takes time if the part is complicated.

1.1.3.2 Graph-based Feature Recognition

Graph based system (Joshi & Chang [1988]), (Sakurai & Gossard [1988]) require matching the feature graph to the appropriate sub-graphs in a solid modeller database.

Graph-based algorithms organize the B-Rep of an object into a graph structure. These graphs can have either faces, edges or vertices as nodes and any of the other two entities as arcs. During the recognition process, these graphs are split into sub graphs using well-developed graph manipulation algorithms. The disadvantage is that extensive preprocessing is necessary to construct the graphs and additional computation is necessary to extract the feature sub graph from rest of the graph.

De Floriani [1989] designed an algorithm for extracting certain classes of shape features of an object from a relational boundary model. The Generalized-Edge-Face-Graph (GEFG) (Ansaldia [1995]) describes objects with multiple shells and multiple connected faces. Feature extraction and classification methods partition the GEFG into sub graphs corresponding to the bi-connected and tri-connected components of the associated face-edge graph. DeFloriani also uses the Object Decomposition Graph (an acyclic directed graph) to describe the processes of decomposition done on the GEFG and to provide an unambiguous, hierarchical description of the global shape of the parts.

Joshi [1988] represents parts and definitions of features by an Attributed Adjacency graph (AAG). An attribute value (zero or one) is assigned to each arc depending on the convexity or concavity. The graphs are stored as adjacency matrices internally. Joshi assumes that "a face that is adjacent to all its neighboring faces with convex angles does not form part of the feature" based on this assumption the algorithm decomposes the AAG into sub graphs by deleting the nodes, which are only connected by convex edges. These sub graphs are further analyzed to determine the feature type. Sakurai [1988] uses a dual representation (both CSG and B-Rep) solid modelling systems for his feature recognition research.

Peter [1991] was one of the first who used neural net for feature recognition from 2D drawings. 2D profiles are represented as sets of lines and curves. Profiles are subdivided into a connected loop of edges. Feature is characterized by the triplet (Ci, Ai, Li) where Ci is curvature, Ai is interior angle and Li is arc length for the ith segment. The input to the neural net is a vector of triplets for a given profile.

Prabhakar and Henderson [1992] first used neural net for solid model feature recognition, developed a system called FRENET. He proposed a five layer quasi-neural net. Input was an adjacency matrix and it recognizes features like holes, pockets, and slots. Output was binary.

Despite their popularity, boundary based feature recognition methods suffer from lack of robust algorithms, particularly when feature interaction is present. Lakko and Mantyla [1993] resolved this problem using strategy of incremental recognition of interacting features.

1.1.4 Cellular Decomposition

These methods have been applied to the determinations of machining volumes from the stock and part models. The Boolean difference between the stock and part model yields the total volume to be removed, but it needs to be decomposed into pieces which correspond to specific machining operations.

Sakurai and Chin [1994] used the concept of volume feature and surface feature. Volume feature is defined as a cavity volume whose area-contact faces have one of the sets of topological and geometrical characteristics pre-defined by the user. Surface feature is defined as the set of the opposite faces of the area contact faces of a volume feature. The method has three steps:

Cell decomposition – Decompose the space surrounding the solid model into bounded minimal cells with all the geometric surfaces of the solid model

Volume composition – Compose combinations of cells into volumes.

Volume checking – Check each volume, if its cavity volume and if it has the topological and geometrical characteristics pre-defined for each type of features.

Vanderbrande and Requicha [1994] used feature completion to produce largest volumetric feature that is compatible with the available data. Algorithm starts with B-Rep of model and original stock. Once the trace of particular feature is found it is extended in one or more directions without intruding into the part. Feature completion is a general method to deal with feature interactions. It associates volumes to boundary data, reconnects portions of a feature that were split due to interactions and produces accessibility information.

Shah et al. [1994] developed a volume decomposition method called minimum convex decomposition by half space partitioning to recognize machining features. First a total volume to be removed is found by subtracting the model from the stock. This volume is then decomposed into minimum convex cells by half space partitioning at every concave edge. A method called maximum convex cell decomposition is developed to generate all alternative volume decompositions. The composing volumes are classified based on degree of freedom analysis. Manufacturing knowledge is characterized in fundamental terms, based on the tool and work piece motion by means of algebraic expression. Inverse mapping of decomposed volumes into these expressions determines feasibility of various machining operations.

1.1.5 Feature Mapping

Features are viewpoint dependent, i.e. each application views the same part in a different way. If a design by feature system is used for defining a part, then the geometric model, which is domain-independent, and the feature model, which is domain-dependent, are produced simultaneously. Some early work has been done in determining manufacturing features from dual feature-geometric models.

Shah [1994] classified mapping classes as follows
One-to-one: when no mapping is required (Identity Transformation).

Variant reparametrization: different sets of dimensions are used to define the same feature in different domains.

Discrete aggregation: two or more whole features combine to produce one feature.

Discrete decomposition: one feature decomposes into two or more whole features.

Conjugation: a new feature is synthesized from portions of two or more features.

Even for an application that uses a fixed set of features and receives the design model from a closed set feature modeller, it s necessary to enumerate all mapping situations that may arise, and to specify the mapping procedure, in advance. However, this is a difficult proposition and susceptible to combinatorial explosion.

In light of the above problems, many research groups seem to have independently reached the conclusion that it is necessary to develop a set of feature building blocks that are domain independent, but are at higher level than just geometry.

Crawford and Srikantappa [1994] call this "Intermediate geometry" and discuss it in terms of GT coding. Automated Group Technology coding relies on a complete geometric description of the part, along with the description of the high level manufacturing features such as slots, holes and pockets, and the spatial relationship between these features. The research is focused on a method to perform automatic GT coding that utilizes geometric model of the part, and a formal description of the relationships among the features of the part. The representation includes a set of structural and geometric primitives, along with a methodology for modelling features in the context of interacting and interfeature relationships. This formalism is referred to as the "intermediate geometry" representation. The information represented includes interactions between features and geometric relationships among features. Shah/Rogers have proposed a set of basic relationships and "generic control elements" from which all features are synthesized. The premises of these approaches is that if features are all described in terms of all basic entities (higher level than geometry) it will be easier to transform feature models from one set to another.

1.1.6 Feature Based Modelling

Large multifunctional manufacturing companies often experience problems because of poor communications between divisions. While product design division have sound knowledge about designing the product, they have less expertise in manufacturing and vice-versa. Although designer may generate physically attractive and functional products, the company may not be able to produce, assemble or maintain them.

Companies have begun to investigate ways to conduct a manufacturing analysis of proposed product design to eliminate downstream problems of manufacturing, assembly, maintenance etc. Such design approaches are referred to as Concurrent engineering. While the product design the manufacturing aspects of the product also taken into consideration in this approach. So it assures for the downstream processes of the product.

Computer aided design for manufacture is one way to achieve concurrent engineering. Traditional CAD systems store information about the part's surface, but not information about its component shape features. They are called design-by-surface with feature recognition system. These systems recognize shape features by abstraction from surface or solid models. These systems have limitations both for designing and for manufacturing analysis. First generating the design-by-surface makes the design tedious. Second the problem of recognizing shape features, which may vary in topology and geometry, is generally computationally expensive.

Design-by-features provides a second design approach. In this approach, features are incorporated in the part model from the beginning. Generic feature definitions are placed in a library from which features are instantiated by specifying dimensions and location parameters and various attributes. Researchers have considered two major sub-categories of the design-by-feature approach.

1.1.6.1 Destructive Modelling With Features

This approach was originally proposed by Arbab [1982]. A part model is created by the Boolean subtraction of features from a base stock model. It may be regarded as a subset

of synthesis-by-features approach. As such it must contend with many of the same problems: predefinition of the generic features, methods for positioning, generation of solid models from feature definitions and feature validation. The design and manufacturing plans are simultaneously developed by transforming a base stock model into the final part model through the application of operations that correspond to stock removal. Prototype systems using this approach have been developed at Stanford [1988] and at Purdue [1988].

In the Purdue system, the work piece was always a rectangular block. In the Stanford system, the work piece was an extrusion of any shape (linear sweep). All these systems used a set of predefined features that were subtracted from the base solid. In the Purdue and Stanford systems, process plans were generated and tested with each design change. In the Stanford system a team of expert systems worked concurrently to generate, simulate and verify plans. Expert systems included feature-machining experts, tooling experts and collision checkers. Features were defined by attribute slots encompassing dimensions, tolerances, finish and starting (tool entry) face.

In the Purdue systems, a feature model was a list of instances consisting of two levels of information. The upper level encoded information common to all features, and the lower had specific information. Common data includes position and orientation matrix, pointers to geometric representations, pointers to reference features, and reference handle. Handles were characteristic geometric elements of features representing points and lines of interest. Point handlers were used for positioning and orienting features and for establishing relationships between two or more features. Line handles were used to represent vectors, which included information such as depth of hole or length of slot. Position tolerances were associated with position vectors. The position of a feature and tolerance stickups were derived by procedures that traced through the hierarchical chain of reference features.

1.1.6.2 Synthesis By Features

Systems that allow one to design by adding or subtracting features without a starting base stock come under this realm. The user is provided with the library of features and Boolean operators with which he can model the part. The features can be classified as

negative features such as hole, slots, pockets etc. or positive features such as protrusion, bosses etc. Further some classified them according to the degrees of freedom of the cutter motion. For example, 1-DOF features are made by plunging a cutter along its axis to produce holes; 2-DOF features are made by plunging and sweeping the cutter along a specific trajectory to produce features such as slots; and 2.5-DOF features are made by plunging a cutter and then cleaning a area to produce pocket like features.

Shah and Rogers [1988] have developed the form feature modelling shell called as FMDS, which provided the facility for creating the product database except the actual definition of features. FMDS can be customized by the organization using it to define the features needed by their designers. Thus feature "knowledge" is added by the organization to get a complete feature definition system. Once the customization is complete, designers can start using FDMS to define products.

It provides for instancing of features from the library, positioning and orientation of feature, modification and/or deletion of feature etc. The solid representation of a generic feature is created using a typical set of primitives and sweeps: box, cylinder, wedge frustum, solids of revolution and linear sweeping. Combinations of these solids in a specified language along with a Boolean operator define the feature-producing volume (FPV). Positive volume features (block or boss) result from union and intersections and negative volume features (pockets and holes) are obtained by subtraction.

Feature Solid Modelling Tool (FSMT) [1993] is developed at Huazhong University of Science of Technology, China. In contrast to the common approach of simply attaching manufacturing information to existing geometric models, the feature definition in the FSMT has been greatly augmented. Geometry is always associated with the knowledge of applying it. The knowledge represents the theory and methodology of design and manufacturing, and a CAD/CAM integration strategy. Generalized sweeping is developed as a unified method for defining various features. Through the development of a dual representation schema of a CSG index and a B-Rep index, attributes such as tolerance and roughness can be conveniently assigned to individual surfaces through the CSG model or the B-Rep model. The CSG model and B-Rep model are easily accessible, and the underlying consistency and integrity is guaranteed. The feature definition also provides a sound mechanism for mapping into applications such as FEM-mesh generation, process

planning and NC programming. As a result feature information is directly available to downstream activities and feature extraction or feature recognition is no longer needed.

Chuan-Jun Su et. al. [1995] proposed Euler-operator based approach for efficient and effective form feature encoding and manipulation in feature based design environment. They introduced the concept of enhanced CSG tree of features (ECTOF), which integrates feature model with solid model in a tree structure.

1.2 Objective And Scope Of The Work

An attempt to provide the link between CAD and CAM started from two decades. It initially started with feature recognition. Many researchers dealt with various feature recognition algorithms as described in the preceding section. First they started with ax symmetric or 2.5D prismatic parts or the 2D drawings and worked towards the feature recognition from 3D models. The basic objective was to extract the features from the model. Afterwards research moved on to feature mapping which was the next step towards Computer Aided Process Planning. The main research here was to map the design features to manufacturing features. Each research group came up with their own strategy for feature mapping. Once the design features are described in terms of manufacturing parameters, automated process planning becomes the next task.

There are certain disadvantages with this approach. Even though much work has been done in feature recognition, still it is constrained to part geometry and can't be employed for all types of geometries. Also the first step in any feature recognition system is importing data from some solid modelling software in some neutral file format such as IGES or STEP. Reading such a file format and extracting the information from the file is very tedious work, also the feature recognition algorithm is computationally expensive.

In the stage of feature mapping, one may face the difficulty that the feature may not be manufacturable which occurs quite sometime due to lack of concurrent engineering. This type of situation needs modelling of the part again and penalizes on time and cost both.

Taking these disadvantages into considerations various researchers have proposed design-by-feature or feature based modelling approach. This approach eliminated the need of importing the data from any solid modeller also eliminated the need of feature recognition. A library of features is provided with Boolean operators to model the part. Since the model is created using the features itself the manufacturability of the part is guaranteed also these features can be directly used to downstream processes such as process planning and NC programming. The present work proposes the similar approach as feature-based-modelling as been proposed by Perdue University or Stanford University but with a more easier way of modelling the part. Since each feature is nothing but an instance of the process and can be described in terms of the process and process parameters, the present work proposes "process-based-modelling". In case of design-by-feature or design-by-synthesis method one need to know about the basics of solid modelling such as extrude, revolve or sweep operations and the Boolean operators Union, Subtract and Intersection. The present system is designed by keeping an ordinary user in mind who have little knowledge about the solid modelling and who has basic understanding of machining processes such as turning, facing etc.

For a normal user it will be easier to understand to drill a hole instead of subtract a cylinder from other cylinder. Again with feature-by-design he should first create an instance, position and orient it and then carry out the Boolean operation, which is more tedious, that just making a drill at specified position.

The present work considers only axisymmetric parts. The modelling starts with selecting the original stock and then subsequently specifying the processes he needs to carry out on the stock to convert it into a final product.

The present work has the objectives of

- Implementation of process-based solid modelling approach for the following processes:

  Parting
  Turning

Facing

Slotting

Taper turning

Drilling

Boring

- Create a graphical user interface (GUI) for easy handling of the software

- Provide various viewing features such as wire frame display, scaling the model, translational and rotational controls etc.

- Provides various solid modelling features such as multiple undo, history tree editing etc.

- Provides for I/O operations such as save and open.

- Provides for interfacing with other downstream applications such as exporting the model, exporting the 2D profile for process planning operations etc.

- Incorporates the intelligence for sequencing of operations once the part is modeled

1.3 Organization Of Thesis

Chapter 2 - Elaborates on different modelling schemes and the one that has been implemented in the software. It also enlists the data structure used and the program algorithm.

Chapter 3 - Describes various features provided in the software. It also tells about the viewing controls and I/O operations provided by the software.

Chapter 4 - Discuss the case study in detail.

Chapter 5 - Concludes the work focusing on the future scope.

Chapter 2

System Design and Implementation

The present chapter discusses various modelling schemes and presents why the particular modelling scheme has been used in the software. It also discusses about the program algorithm and enlists the data structure that is being used for implementation.

2.1 Modelling Schemes

There are six major methods of constructing solid models. Instances or parameterized shapes, cell decomposition including spatial occupancy enumeration, sweep representations, constructive solid geometry, boundary representations and wire frame representations. Underlying them are the concepts of graph-based models and Boolean models.

2.1.1 Graph Based Models
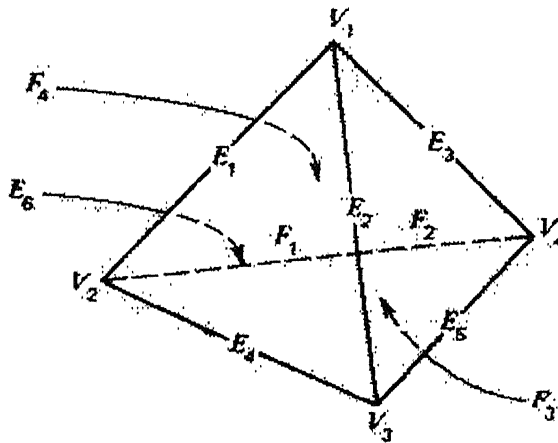
Fig. 2.1 : A Simple Tetrahedron

Fig. 2.2 : Data Structure For Graph Based Model

A geometric model emphasizing the topological structure with data pointers linking together an object's faces, edges and vertices, is a graph-based model. A solid object can be represented as a list of the object's faces and their respective surface equations. The edges of these surfaces are represented as curve equations, with pointers to their end point vertices and adjoining faces. The vertices are represented as list of coordinates, with pointers to the edges meeting at each vertex.

1 = Connectivity
0 = No connectivity
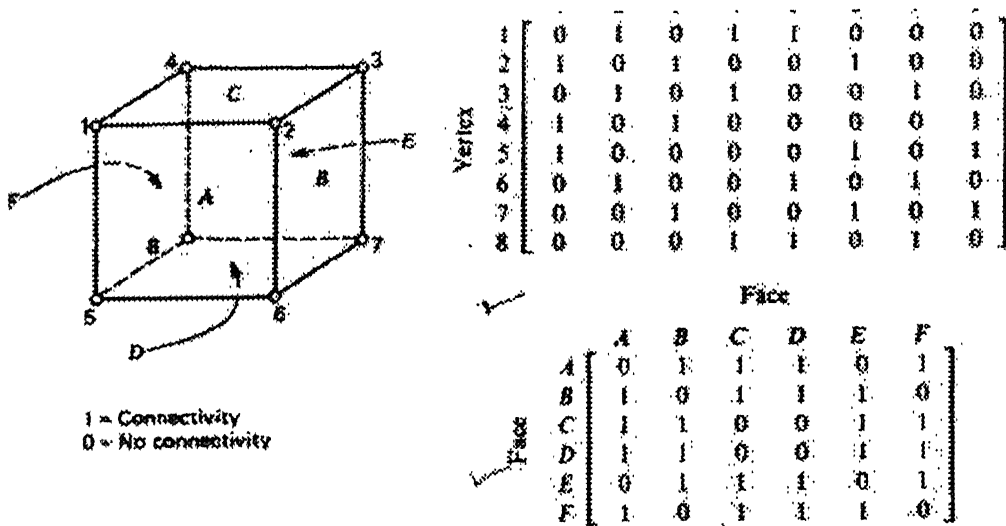
Fig. 2.3 : Connectivity Matrix

There are two kinds of information – the pointers defining the topology or connectivity between vertices, edges and faces and numerical data defining curve and surface equations and vertex coordinates as shown in figure 2.2 for the tetrahedron of figure 2.1.

The topological information can also be stored in the form of a connectivity matrix or an adjacency matrix as shown in figure 2.3.

2.1.2 Boolean Models

If a solid object is represented by the Boolean combination of two or more simpler objects, then the representation is a Boolean model. If A, B and C denote solids and if C = A<OP>B, where <OP> is any regularized Boolean operator then A<OP>B is the Boolean model of C. It is a procedural model.
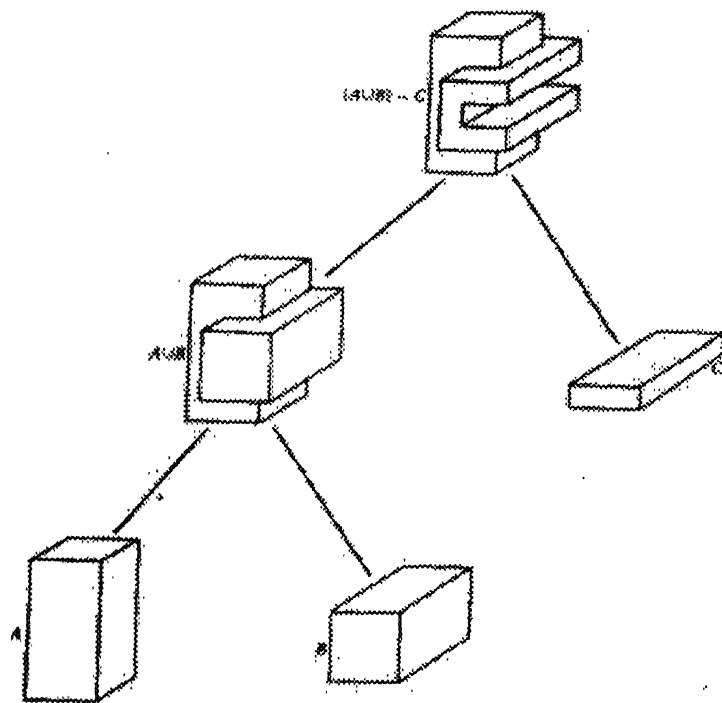
Fig. 2.4 : Boolean Model For D = (A U B) - C

Figure 2.4 shows the Boolean model of D = (A U B) – C. This Boolean statement defining D says nothing quantitative about the new model it creates; it only specifies the combination of primitive solid constituents. It does not tell the coordinates of the vertices

of the new solid or about its edges, faces. All we know about D is how to construct it. So it's a procedural representation or unevaluated model. If we want to know more then we have to evaluate the Boolean model. We have to compute intersections to determine new edges and vertices. We have to analyze the connectivity of these elements to determine topological properties. Boolean operators are Union, Difference and Intersection as shown in figure 2.5.
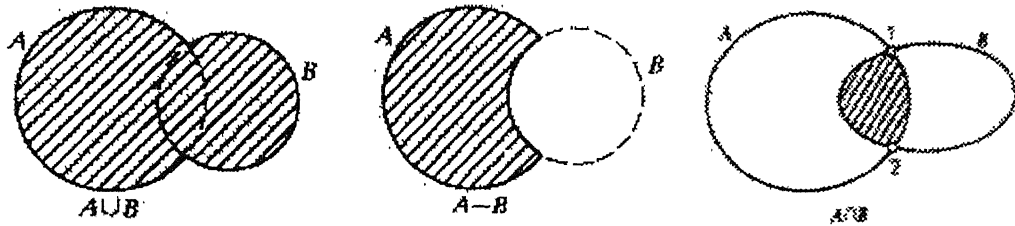
Fig. 2.5 : Boolean Operators

2.1.3 Instances And Parameterized Shapes

A direct way of defining a new shape is a simple linear transformation of an existing one. Consider a unit cube as shown in figure 2.6. It can be described by its width, length and height. There are several ways of transforming it into new shape by using scaling operators. Equal scaling in all three dimensions will create new cubes while differential scaling will create new rectangles. Each new cube or rectangle solid is a particular instance of the original cube.

Instancing is not limited to simple shapes. Wherever we can define the object in terms of different parameters, we can create instances of that object. It can be clear from the figure 2.6.
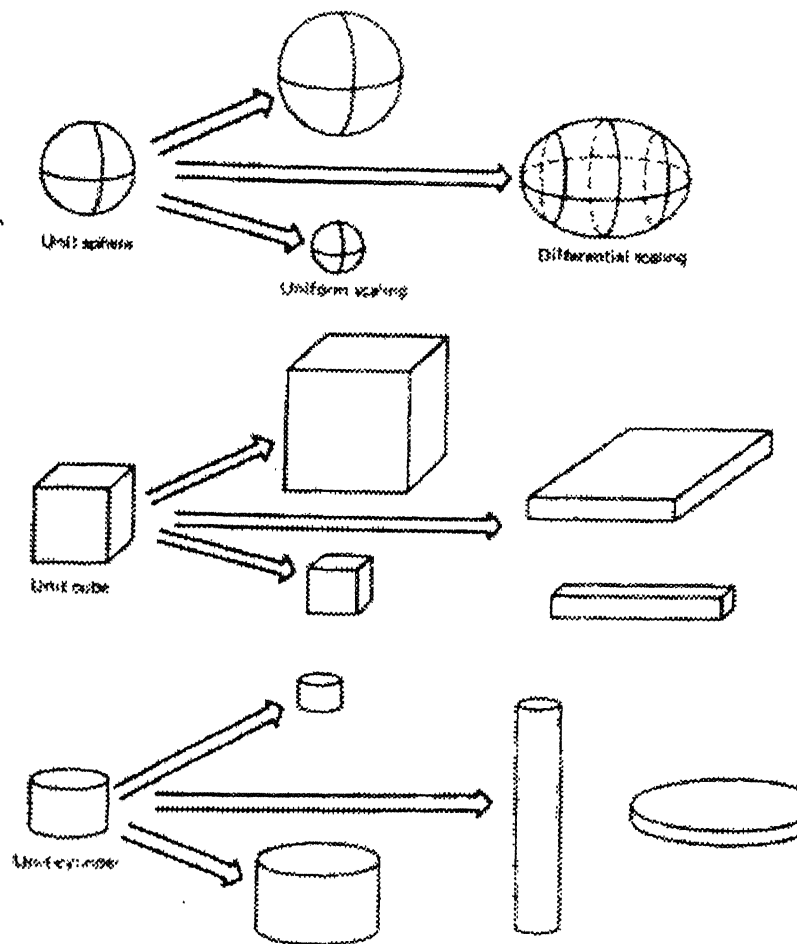
Fig. 2.6 : Primitive Instancing

2.1.4 Cell Decomposition and Spatial Occupancy Enumeration

Take a coffee mug and decompose it into separate pieces as shown in figure 2.7 such that each piece of final decomposition is easier to describe than the original mug.

This process is cell decomposition. Any solid can be represented as the sum or union of a set of cells into which it is divided. The reason for using cell decomposition is that the total object may not be amenable to representation, but its cells are. There are many ways of decomposing a solid into constituents, none are unique but all are unambiguous.
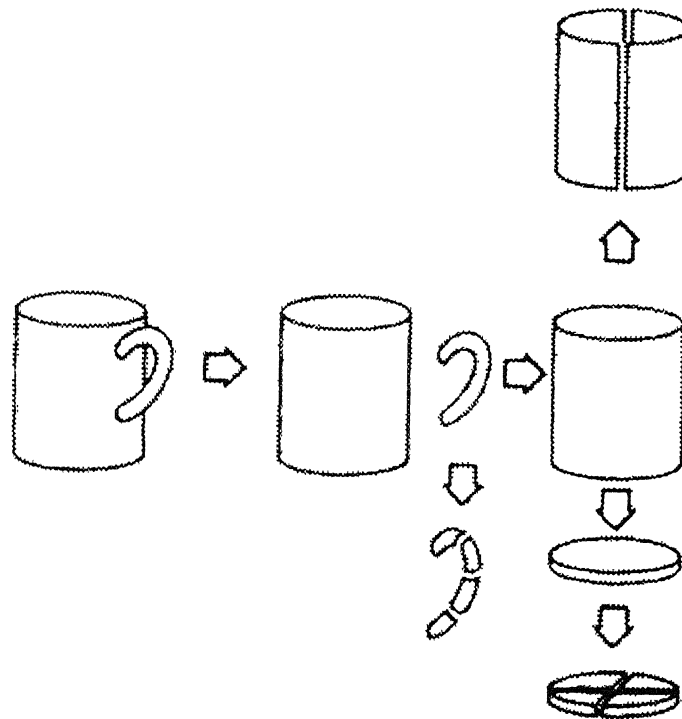
Fig. 2.7 : Cell Decomposition

Spatial Occupancy Enumeration is a special case of cell decomposition where cells are cubical in shape and located in fixed spatial grid. As the size of the cube decreases, this method approaches the representation of solid body as a set of contiguous points in space Defining a solid using spatial occupancy enumeration requires a convenient way of representing this set of cubical cells. One way is simple listing coordinates of centers of the cells; a solid is thus a set of adjacent cells. Cell size determines the resolution of the model. Smaller the size more the resolution, but more the storage required.

Quadtrees and their three dimensional analogue Octrees, suggests a way using spatial occupancy enumeration more efficiently. Quadtree is based on recursive subdivision of square array into quadrants. Each node represents a square region on the plane and has four descendants. If the height of tree is n then maximum potential array size is $2^n \times 2^n$. Octree is based on recursive subdivision of cube into octants or eight cubical regions. Each node, which is not a leaf node, has eight descendants. For a tree of height n the potential size of array is $2^n \times 2^n \times 2^n$.
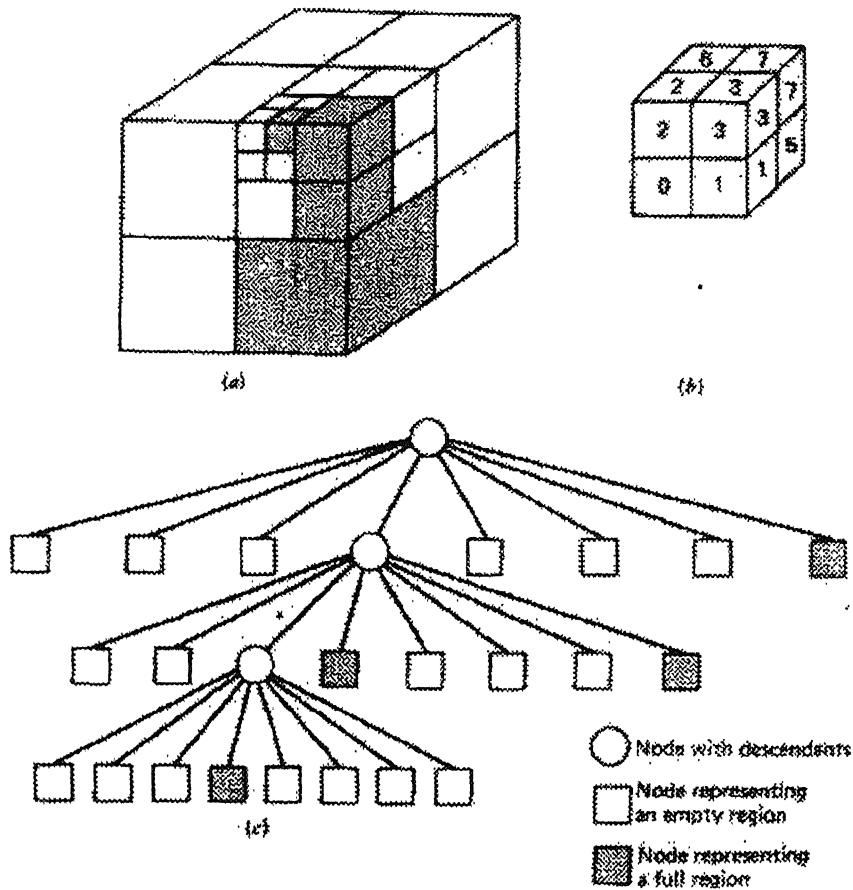
Fig. 2.8 : Octree Encoding

2.1.5 Sweep Representation

Sweep Representations are based on notions of moving a point, curve or surface along some path. The locus of points generated by this process defines a one-, two- or three-dimensional object.

For solid modelling two ingredients are required the directrix and the generatrix. The directrix is a trajectory, which is an analytically definable path and generetrix is an object that may be curve, surface or solid and is moved along the trajectory. Two principle types of trajectories are depicted – translational and rotational.
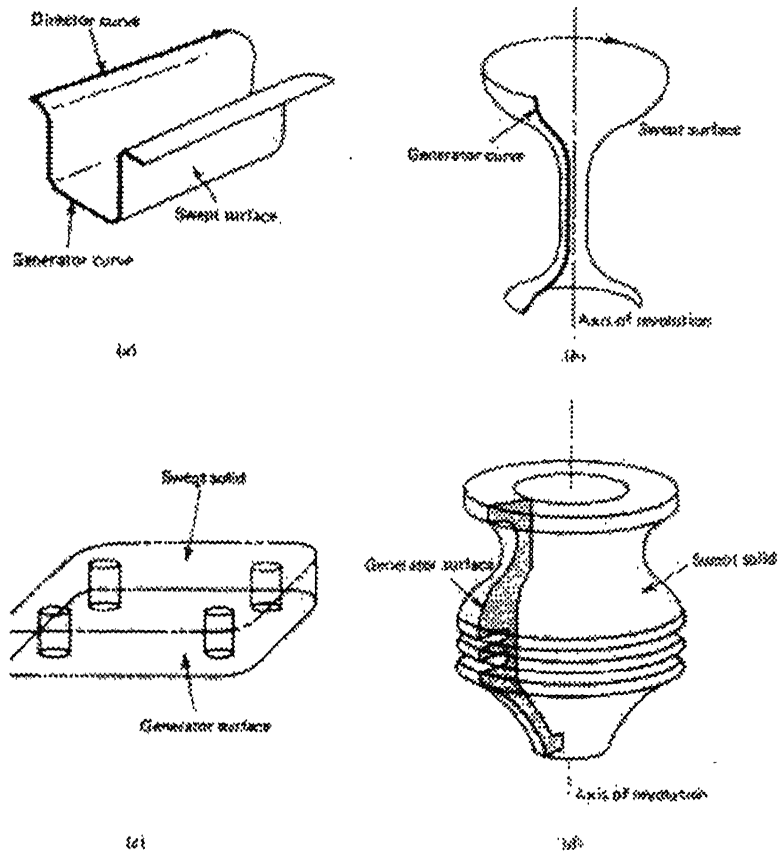
Fig. 2.9 : Various Sweep Generated Models

2.1.6 Constructive Solid Geometry

Constructive Solid Geometry (CSG) is a term for modelling methods that define complex solids as compositions of simpler solids (primitives). Boolean operators are used to execute the composition.

CSG representations of objects are ordered binary trees whose leaves or terminal nodes are either primitives or transformation data for rigid-body motions. The non-terminal nodes are either regularized Boolean operators (Union, Difference or Intersect) or rigid body motions (translation and/or rotation) that operate on their two sub nodes.
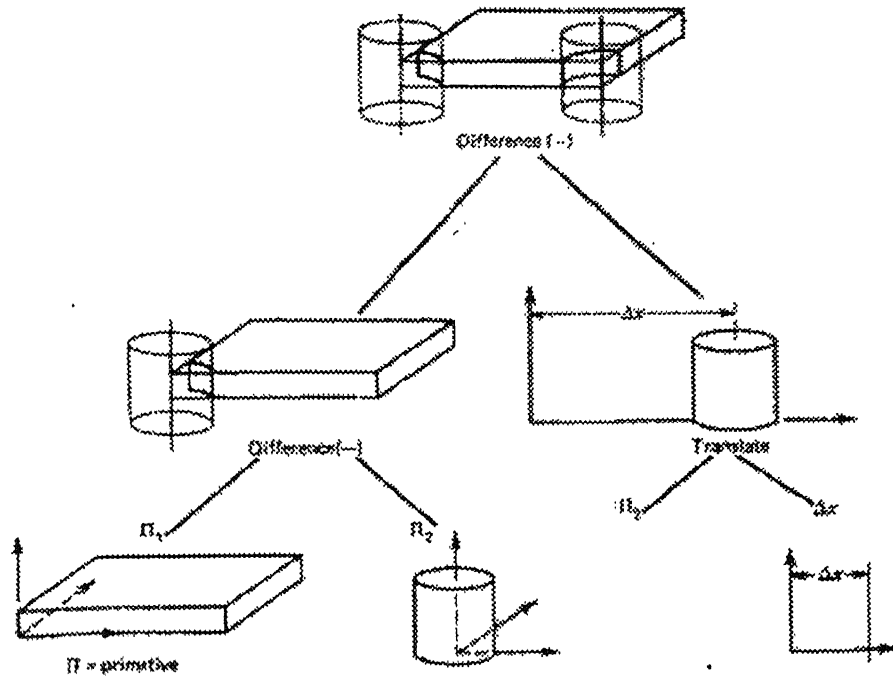
Fig. 2.10 : CSG Representation

Each sub tree of a node (not a transformation leaf) represents a solid resulting from the combination and transformation operations indicated below it. The root represents the final object.

2.1.7 Boundary Representation (B-Rep)

In B-Rep the solid is represented by its boundary surfaces, which encloses the solid, which in turn is represented by their boundary curves, which defines the surface and which in turn is represented by its end points which defines the curve.

Thus B-Rep is a graph-based model as discussed earlier.

2.2 Modelling Scheme Implemented

The present software uses Cell Decomposition for constructing solid model. Only CSG based scheme cannot be used since it requires calculating of intersection of surfaces and curves and vertices to represent the solid. B-Rep is more complicated to use in the present

context since the software deals specifically with axisymmetric parts. Spatial Occupancy Enumeration takes more data storage and resolution depends on the size of the cells. Hence cell Decomposition provides a better option.

The basic primitive in the software is a frustum of a cone which is bounded by two end faces viz. front face and back face and a conical face. The basic parameters used to define this primitive are the inner radius and outer radius for the two end surfaces. So it can represent both cylindrical cell and conical cell with or without a hole. Also the model dictates the number of cells at any stage of modelling and it is not of the fixed size as in the spatial occupancy enumeration. Thus it reduces the memory requirements considerably. So the model at any stage would be a union of all the cells, which may be cylindrical or conical.

2.3 Data Structure Implemented

Software maintains three data structures

- Process Data Structure (PDS)
- Model Data Structure (MDS)
- Feature Data Structure (FDS)

2.3.1 Process Data Structure

This is a basic data structure for the software and other data structures and operations like undo, history list editing etc. are all done with respect to process data structure, thus truly making it a *process based modeller*.

It is a sequential linked list which stores the processes in a sequential order in which user has performed the operations. Each node of the link list represents one process and stores the corresponding process parameters.

28
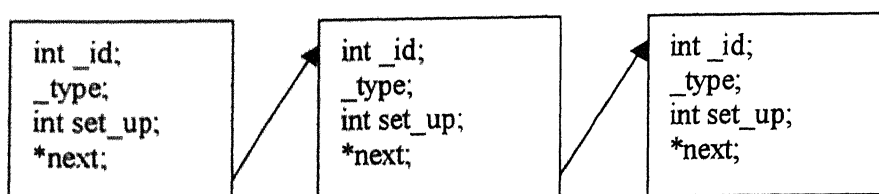
A pseudo code of the process data structure is as follows.

```
struct _process{
        int _id,
        _process_type _type;
        int set_up ;
        struct _process *next ;
};
typedef struct _process _process ;

typedef union {
                sawing _s ;
                turning _t ;
                facing _f ;
                necking _n ;
                taper_turning _tt;
                drilling _d;
                boring _b ;
} process_type;
```

Individual process structures are as following:

```
typedef struct {
        float z_location;
        float cutting_speed;
        int set_up ;
}sawing;

typedef struct {
        float cutting_speed;
        float depth_of_cut;
        float feed;
```

```c
        float z_location,
        float length_to_cut;
        float final_dia;
        int set_up ;
}turning;

typedef struct {
        float cutting_speed;
        float depth_of_cut;
        float feed;

        float z_location;
        float length_to_face;
        int set_up ;
}facing;

typedef struct {
        float cutting_speed;
        float depth_of_cut;
        float feed;
        float len_of_parting;

        float z_location;
        float final_dia;
        int set_up ;
}necking;

typedef struct {
        float cutting_speed;
        float depth_of_cut;
        float feed;

        float z_location;
        float length_to_cut;
```

```
        float small_dia;
        int set_up ;
}taper_turning;
typedef struct {
        float cutting_speed;
        float depth_to_cut;
        float feed;
        float tool_size;

        float z_location;
        float final_dia;
        int set_up ;
}drilling;


typedef struct {
        float cutting_speed;
        float depth_to_cut;
        float feed;
        float tool_size;

        float z_location;
        float final_dia;
        int set_up ;
}boring;
```

2.3.2 Model Data Structure

This is used for storing cells information that composed the whole model in the form of
doubly linked list having two pointers viz. one pointing to the next cell and other pointing
to the previous cell. Two end faces and conical/cylindrical face define each cell. The cell
may be cylindrical or conical with or without a hole.
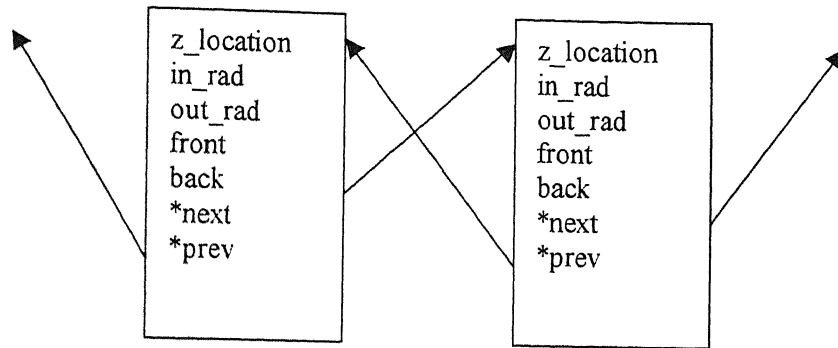
The pseudo code for the cell structure is as follows.

```
struct cell{
        float z_location ;
        float in_rad ;
        float out_rad ;
        int front ;
        int back ;
        struct cell *next ;
        struct cell *prev ;
};

typedef struct cell cell ;
```

2.3.3 Feature Data Structure

This data structure is used to store the features created in each operation in 2D form as a sequential linked list, each node representing the feature removed. These when rotated about the axis of the part will represent the 3D volumes to be removed from the original stock to produce the model.

Pseudo code of the feature data structure is as follows.

```
struct feature{
```

```
        point *p ;
        struct feature * next ;
        int no_of_pt ;
        int set_up ;
};
```

```
typedef struct feature feature ;
```

```
typedef struct {
        float x ;
        float y ;
} point ;
```
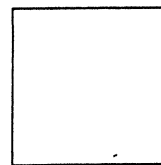
2.4 Software Algorithm
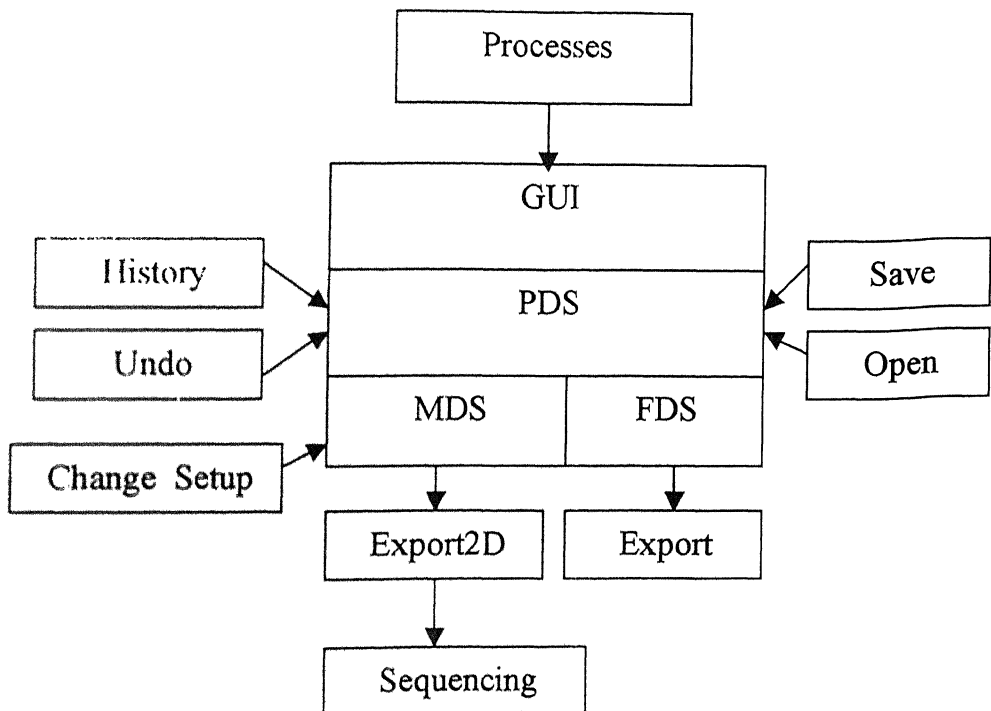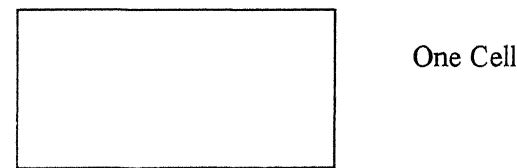

2.11 : Schematic of System Architecture

The modelling starts with the selection of a stock by specifying its length and choosing the diameter from the available list and they have been assigned as global variables. This step will initialize the model data structure (MDS) with two nodes in a doubly linked form comprising one single cell, the process data structure (PDS) not being initialized at this stage.

As the user does the operations on the stock, one node is added to the PDS with the user specified parameters. Once the PDS is updated, the MDS and the FDS are created based upon the PDS. Thus at each stage MDS is being deleted and recreated. It does penalize on the computation time, but it minimizes the memory storage and also provides with many other features to implement smoothly like multiple undo, history list editing etc.
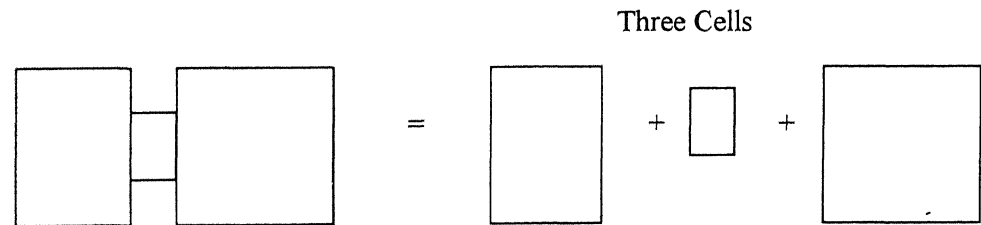
When an undo operation is performed, the last node in the process link list is deleted. Whenever one makes some changes in the history list, that change is reflected in the PDS. Once the PDS is updated, the MDS is rebuilt by traversing the processes' link list from the start node and depending upon the process-id, cells are added to the model data link list

For example, when the process is slotting/necking the original cell is split into three cells as shown below.

Starting MDS

One Cell

After Slotting Operation

Three Cells

The modelling starts with the selection of a stock by specifying its length and choosing the diameter from the available list and they have been assigned as global variables. This step will initialize the model data structure (MDS) with two nodes in a doubly linked form comprising one single cell, the process data structure (PDS) not being initialized at this stage.

As the user does the operations on the stock, one node is added to the PDS with the user specified parameters. Once the PDS is updated, the MDS and the FDS are created based upon the PDS. Thus at each stage MDS is being deleted and recreated. It does penalize on the computation time, but it minimizes the memory storage and also provides with many other features to implement smoothly like multiple undo, history list editing etc.

When an undo operation is performed, the last node in the process link list is deleted. Whenever one makes some changes in the history list, that change is reflected in the PDS. Once the PDS is updated, the MDS is rebuilt by traversing the processes' link list from the start node and depending upon the process-id, cells are added to the model data link list.

For example, when the process is slotting/necking the original cell is split into three cells as shown below.

Starting MDS

One Cell

After Slotting Operation

Three Cells

Certain operations will change only the cell attributes like drilling or boring which changes the inner radius of the cells.

In the Change Set Up operation, the MDS is entirely reversed so that the original start and end cells become the end and start cells respectively. One field named set_up is maintained in each process structure, which stores the information about the set up and is a binary field. The toggling of set up is reflected by the different status of this variable for two consecutive processes.

2.5 Sequencing Algorithm

Machining sequencing is applying the intelligence to identify the correct machining sequence to produce the part taking into consideration the time and cost and other manufacturing attributes. The software provides this feature for direct interface to the downstream applications such as process planning, NC programming etc.

Since the algorithm is only for rotating parts, the input is the 2D profile of the part. Algorithm works in the following stages.

Reduce the original stock to the bounding rectangle of final model by parting/facing and turning. Its the first operation since we have to create the reference surface.

Volume to be removed by turning

Volume to be removed by Parting/facing.
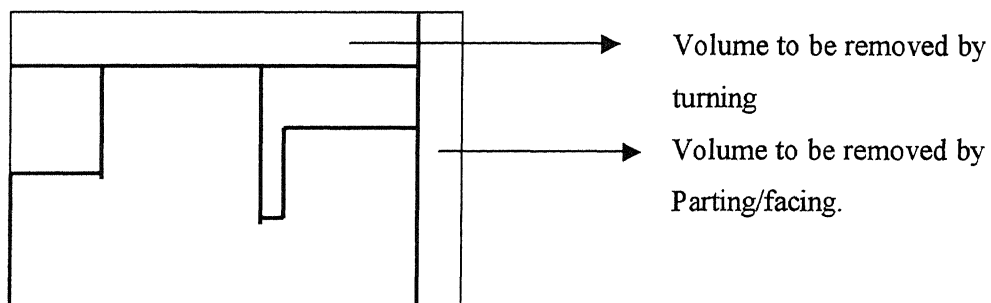
Find the Y-Max and divide the profile into the Right and Left side.
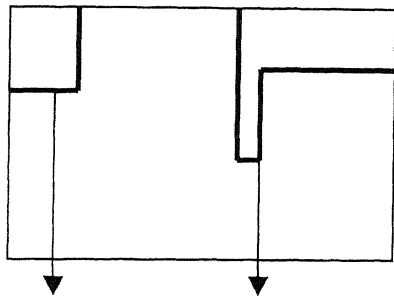
Left Side   Right Side

Apply sequencing for the Right and Left side separately.

Replace the taper surface by taking one virtual point.

Insertion of one virtual point to remove the taper.
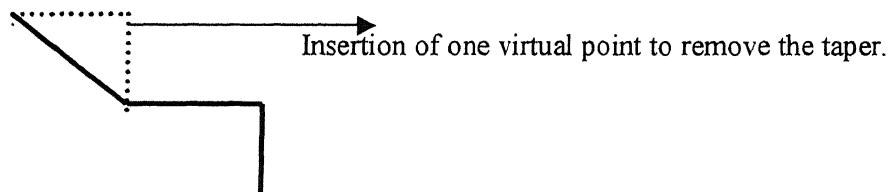
Travel from outside towards the axis. A scan line is taken to check for the intersection with the model, which gives the starting and end points.

If the distance between starting and end point is less than the slotting limit, go for slotting operation else go for turning operation.

Machine for tapered surfaces if any.

Machine for drilled holes if any, starting from the hole with smaller diameter.

The implemented code is not exhaustive to take care of all the degenerate cases, but is an attempt towards compatibility with the downstream applications such as making the process plan and NC programming.

Chapter 3

Software Features

The present chapter discusses about various features of the software. It gives brief introduction of different controls and toolbars and their functionalities.

The entire software is designed on ANSI C compiler compatible with both UNIX and WINDOWS environment having an in-built OpenGL support. For modelling purpose OpenGL and GLUT libraries are used whereas GLUI library has been used for developing the Graphical User Interface (GUI).
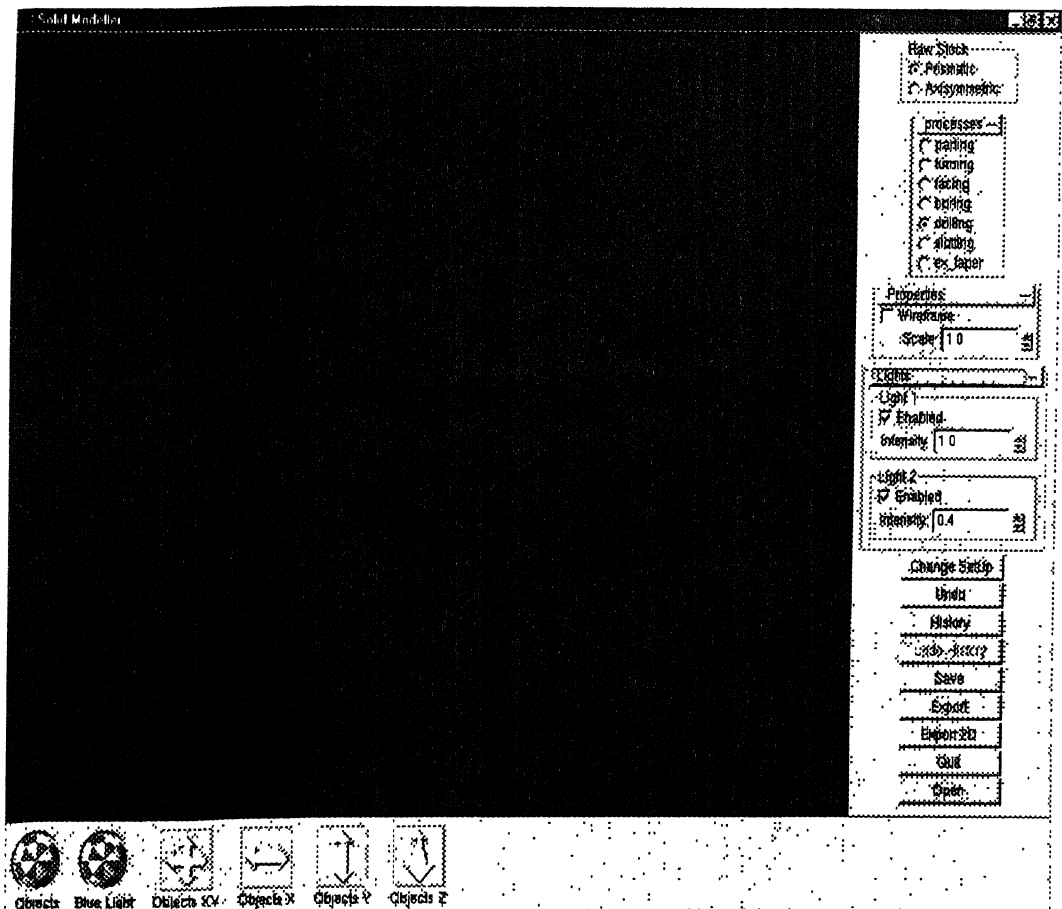
Fig. 3.1 : Graphical User Interface

3.1 Stock Selection Panel
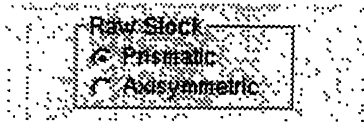
Fig. 3.2 : Stock Selection Panel

This particular module supports only axisymmetric part. When radio button is clicked it asks for the stock diameter, stock length and stock material. The stock diameter is selected from the available database. This database can be modified by modifying the database file "stock.dat" which is an ASCII file containing available stock diameters.
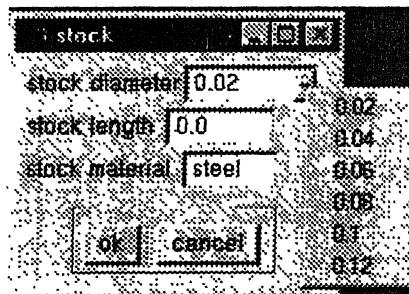
Fig. 3.3 : Stock Selection Dialog Box

Format of the file is as below

stock.dat

6   # No. of bar stocks available

0.08   # Diameter of the stock.

0.1

0.12

0.14

0.16

0.2

3.2 Process Selection Panel
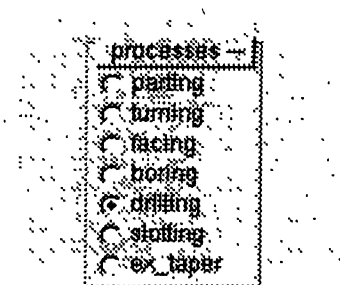
Fig. 3.4 : Process Selection Panel

Software provides for following processes. Each Process asks for the machining parameters and geometric parameters.

3.2.1 Parting
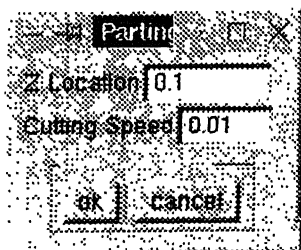
Fig. 3.5 : Parting Process Parameters

3.2.2 Turning
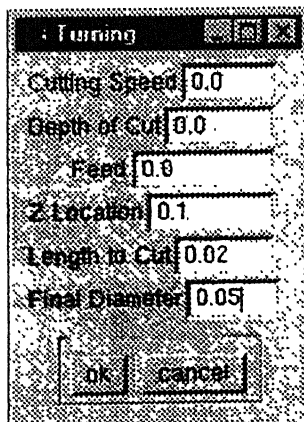
Fig. 3.6 : Turning Process Parameters

3.2.3 Facing

Fig. 3.7 : Facing Process Parameters

3.2.4 Boring

Fig. 3.8 : Boring Process Parameters

3.2.5 Drilling

Fig. 3.9 : Drilling Process Parameters

3.2.6 Slotting

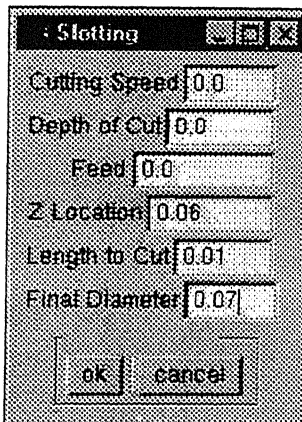**Fig. 3.10 : Slotting Process Parameters**

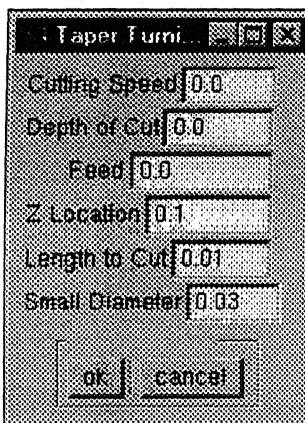### 3.2.7 External Taper

Fig. 3.11 : External Taper Turning Process Parameters
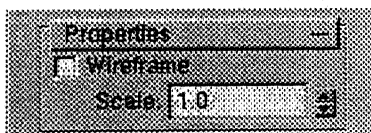
3.3 Properties Panel

Fig. 3.12 : Properties Panel

3.3.1 Wireframe

It provides for wireframe or solid model.

3.3.2 Scale

It provides scaling of the model both scale up and scale down.
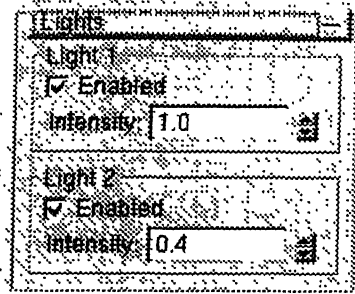
3.4 Lighting Panel

Fig. 3.13 : Lighting Panel

In the software two lights situated at two different points have been provided. Users can enable/disable one of the lights or he can change the intensity of the lights using the controls.

3.5 Change Set Up

With this button setting is reversed.

3.6 Undo

The software provides the unique facility of multiple undo. User can undo as many operations as he done on the stock. This is possible because of Process Based Modelling.

3.7 History

History List Editing is one of the powerful features provided in the software. Almost in any commercially available software one can edit the history tree. But in case of commercially available solid modellers tree is made up of primitives at the leaves and Boolean operations at the nodes. In the present software the history list is made up of processes as the nodes.
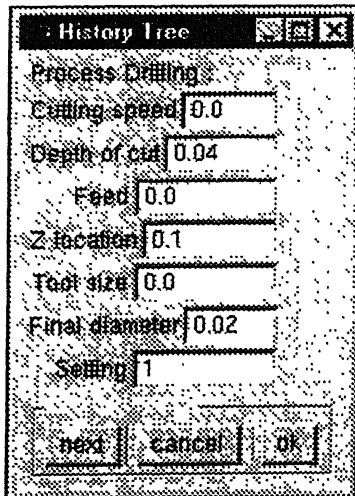

Fig. 3.14 : History List Dialog Box

"next"   button displays the next process with the associated parameters.

"cancel"   button closes the dialog box

"ok"   Does the required modification.

3.8 Undo History

History Tree Editing does not always ensures correct modelling and it may produce absurd result sometimes. User can reject such modifications using this feature.
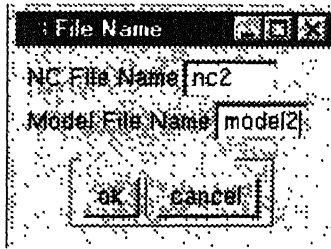
3.9 Save

Fig. 3.15 : File Save Dialog Box

Mere modelling never serves the objective, but the model needs to be stored. Every software has its own data format for storing viz. AutoCAD provides dxf format, IDEAS provides mf1 and mf2 formats. Likewise this software has its own data storage format. It saves the process data structure information in the NC file and modelling data structure information in the model file.

NC File Format is as follows

Selected stock

Stock material _____
Stock diameter = _____
Stock length = _____

Process 1 _____:

Process Parameters:

Cutting speed = _____
Depth of cut = _____
Feed = _____
Z Location = _____
Length to cut = _____
Final diameter = _____
Setting = 0   # stores the setting

Process 2 _._____:

Process Parameters:
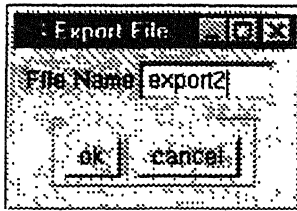and so on .....

44

3.10 Export

Fig. 3.16 : File Export Dialog Box

Almost every commercial software supports the standard neutral file formats like IGES or STEP to export the data from one software to other. Without this portability the scope is very limited to that particular software. An effort is made in this software to export the data in one neutral file format but it provides its own ASCII file, which can be read into other CAD software using their script. For example in AutoCAD it can be read using AutoLISP, in IDEAS it can be read using Open Architecture of IDEAS, in Imageware one can write a script in Scoll to read this file.

Export File Format is as follows

100   # Axis of Rotation
        # Starting Point
        # End Point
10   # Initial Stock
        # Points Representing 2D profile of the stock
0   # Feature
        # Points Representing 2D profile of the feature
1000   # End of File.

From Axis of Rotation and 2D profile one can generate the Initial solid stock and Features and writing for the Boolean operations will produce the final part. So it preserves the history and feature information as well.
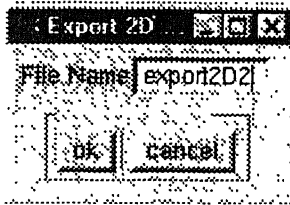
3.11 Export2D

Fig. 3.17 : Export 2D Profile Dialog Box

This feature is provided to use the software for further downstream processes like sequencing, process planning and NC-code generation for Axisymmetric parts.

It exports initial stock diameter, stock length and the 2D profile of the final model. From this much information one write the code for sequencing, process planning and NC-code generation. This is an attempt to make the software as a complete module.

File Format is as follows

Stock diameter
Stock Length
Number of points
Point co-ordinates

.

.

3.12 Sequencing

It generates the machining operation sequence for minimum set up and so time and cost.

The file format is as follows
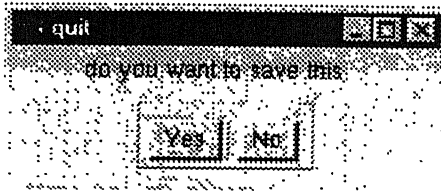
Machining operation
Start point
End point

3.13 Quit

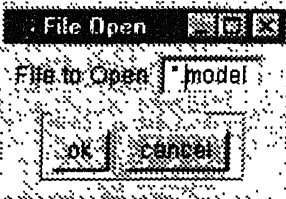**Fig. 3.18 : Quit Dialog Box**

It asks for the confirmation.

3.14 Open

Fig. 3.19 : File Open Dialog Box

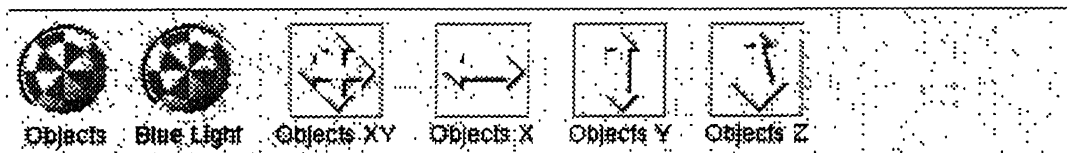Opens the Model File.

3.15 Viewing Controls

Fig. 3.20 : Viewing Control Panel

It provides following controls

Rotation of Object

Rotation of the Light Source

XY-Translation of the Object

X-Translation of the Object

Y-Translation of the Object

Zoom In/Out

Chapter 4

Case Study

This section illustrates a case study of a solid model and shows some other examples modelled using this software.
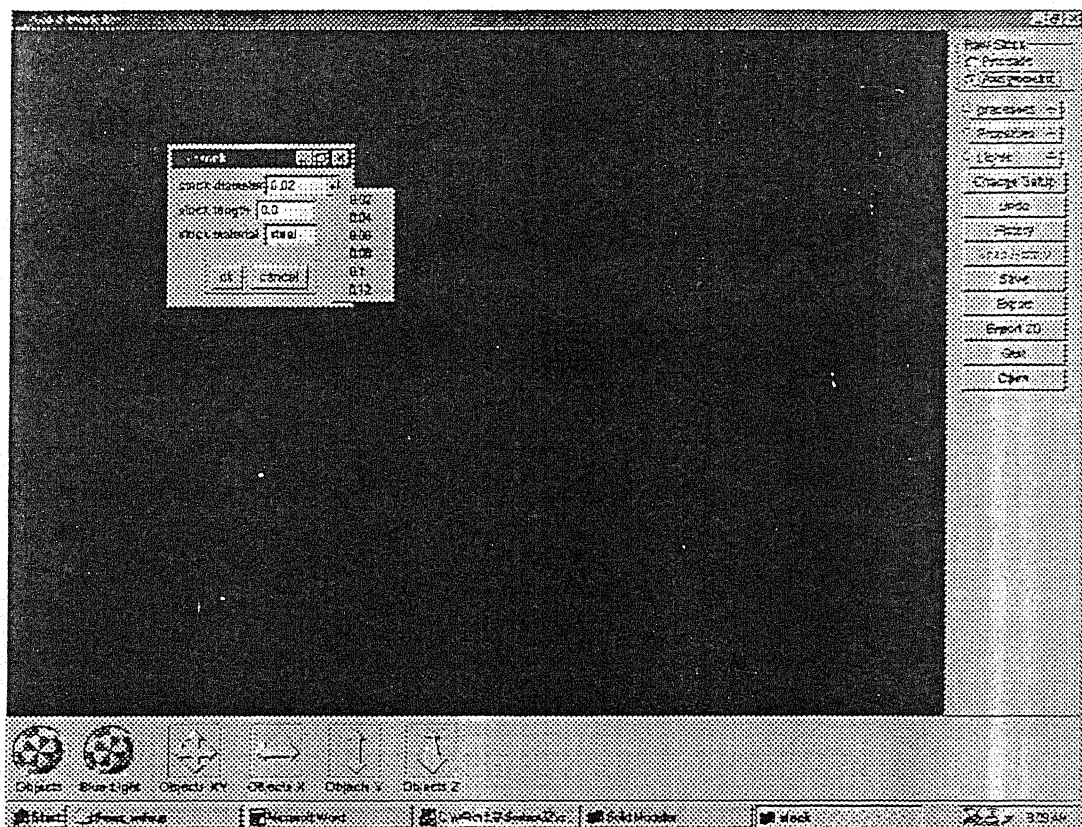
- **Selecting the Stock**

Fig. 4.1 : Selecting The Stock

Values chosen are

Stock Diameter : 0.1
Stock Length   : 0.1

Stock Material : steel

Stock axis is aligned with the Z- axis and left end of the stock is at the head stock center. X and Y axis are according to lathe conventions.

- **After Stock Selection and Specifying Turning Parameters**
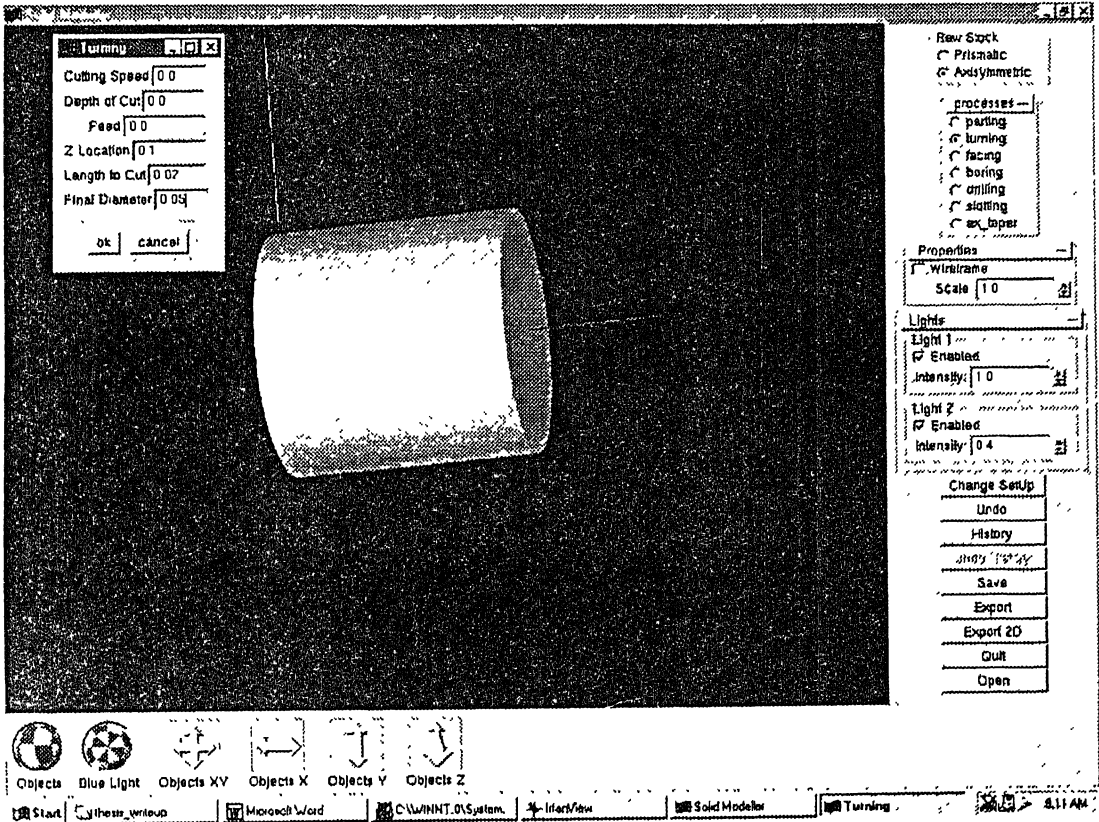

Fig. 4.2a : Stock Selected and Specifying Turning Parameters

Process Parameters

Cutting Speed : 0.01

Depth of Cut : 0.001

Feed : 0.001

Z Location : 0.1

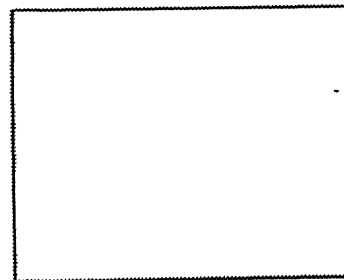Length of Cut : 0.02

Final Diameter: 0.05

Fig. 4.2b : 2D Profile

Machining is done always from the right end as is done on a conventional lathe center.

Z location denotes the starting point, Length of Cut denotes the total length of turning and Final Diameter denotes the Diameter after tuning is complete.

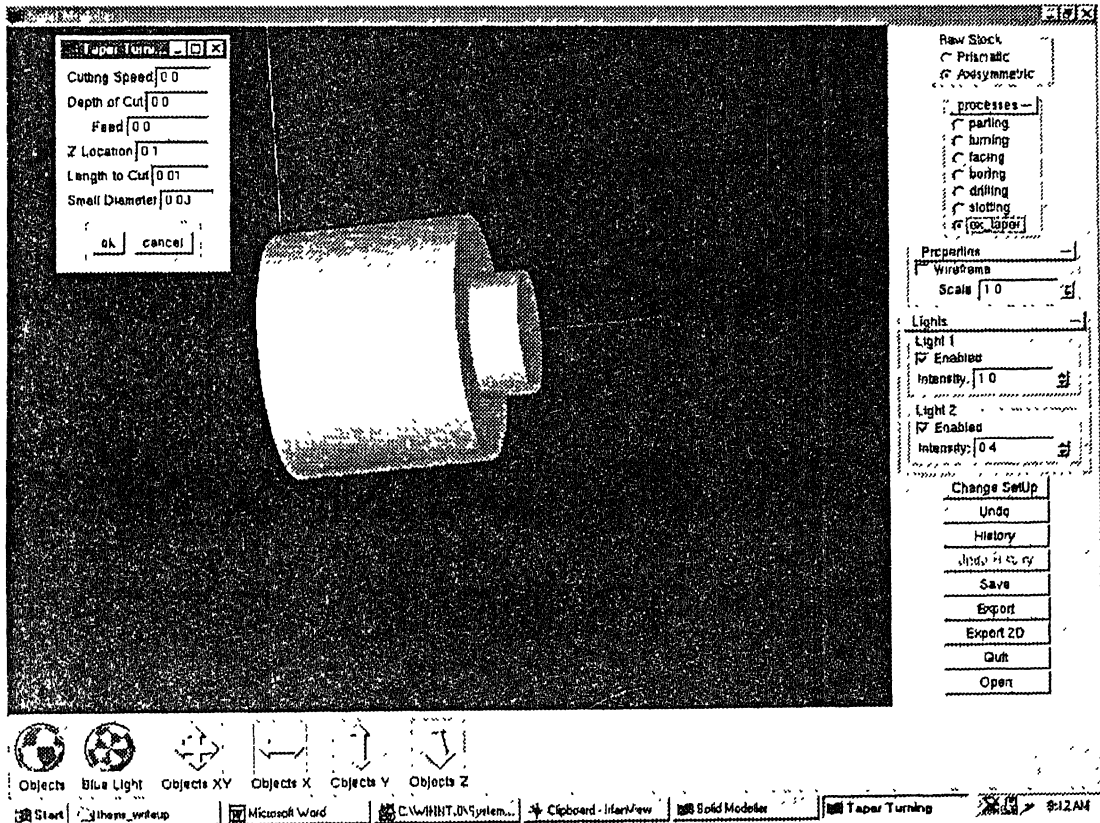- **After Step Turning and Specifying Parameters For Taper Turning**

Fig. 4.3a : Step Turning and Specifying Taper Turning

Process Parameters

Cutting speed : 0.001

Depth of Cut : 0.001

Feed : 0.001

Z Location : 0.1

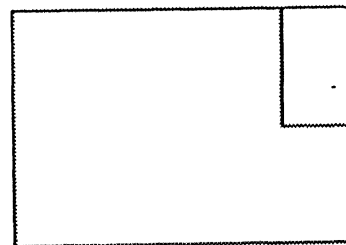Length of Cut : 0.01

Small Diameter: 0.03

Fig. 4.3b : 2D Profile

Angle of taper is not necessary since the length of cut and small diameter defines it. Also the big end diameter is taken as the diameter of the section at the length of cut starting from Z location.

- **After Taper Turning and Specifying Parameters For Slotting**
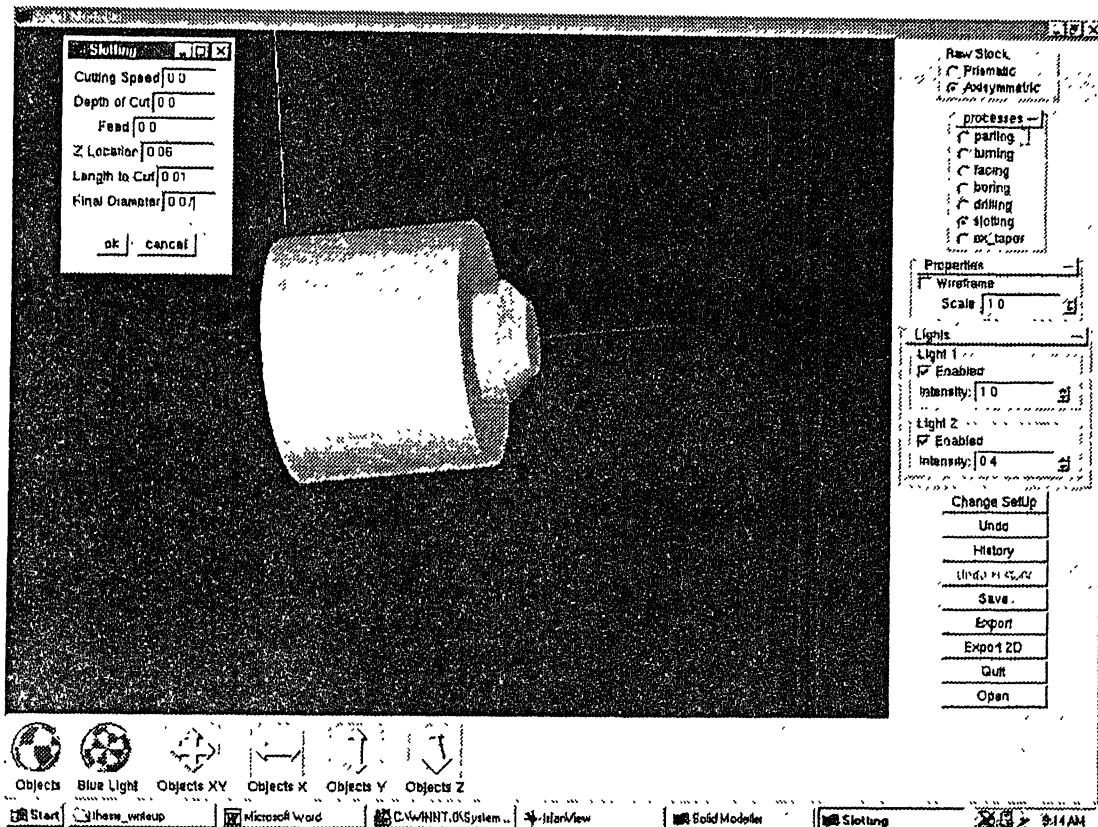

Fig. 4.4a : Taper Turning and Specifying Slotting Parameters

Process Parameteres

Cutting Speed : 0.001

Depth of Cut   : 0.001

Feed   : 0.001

Z Location   : 0.06

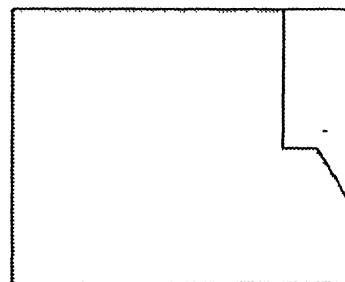Length of cut : 0.01

Final Diameter: 0.07

Fig. 4.4b : 2D Profile

Z Location defines the starting point for the slot and Length of Cut defines the width of the slot. Slotting is generally required when no free face is available for direct turning and slot width is lesser. Final diameter of the slot is only required since the starting diameter is the diameter before the operation at the starting and end point.
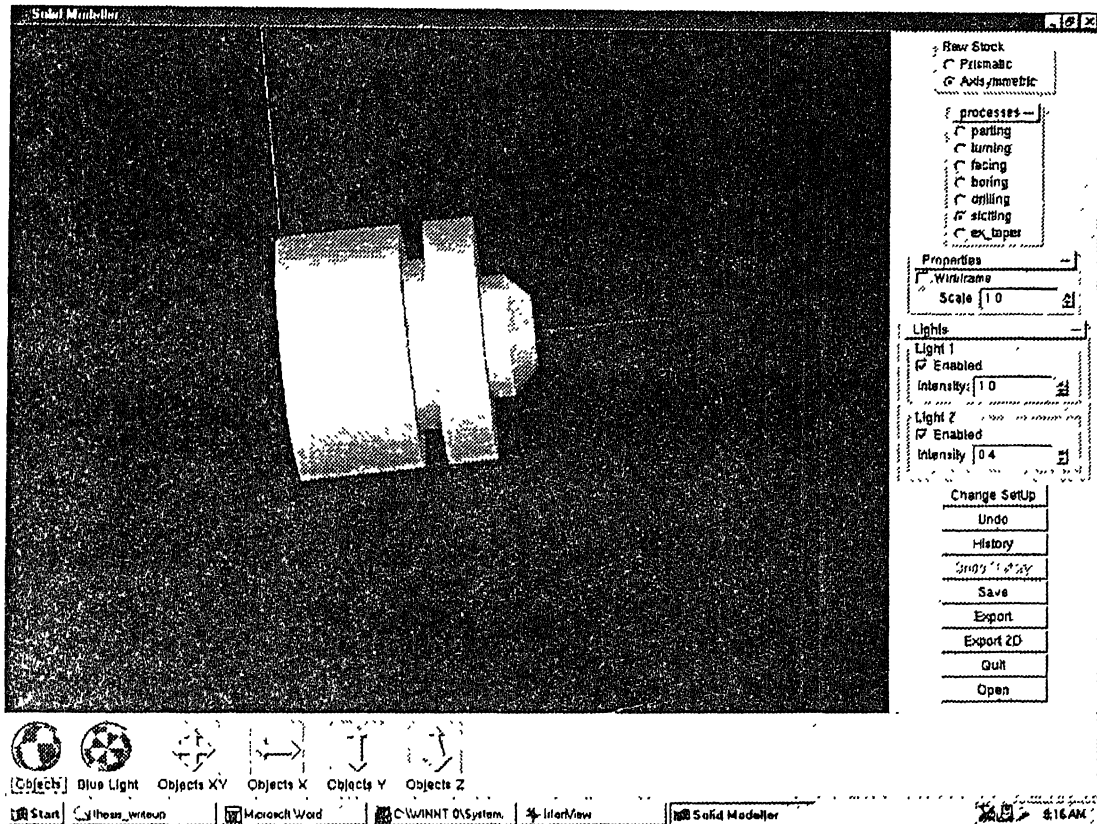
- **After Slotting**

Fig. 4.5a : Slotting

For drilling a hole on the other face we have to change the set up i.e. model will be rotated about the X-axis by 180 degrees.

For this purpose **Change SetUp** button is provided. It reverses the data structure to reflect the changed setting.
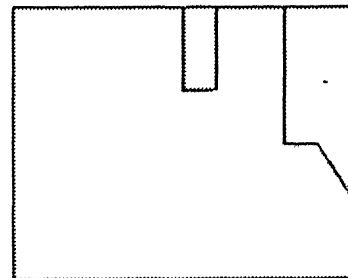
Fig. 4.5b : 2D Profile

- **After Change Set Up and Specifying Parameters For Drilling**
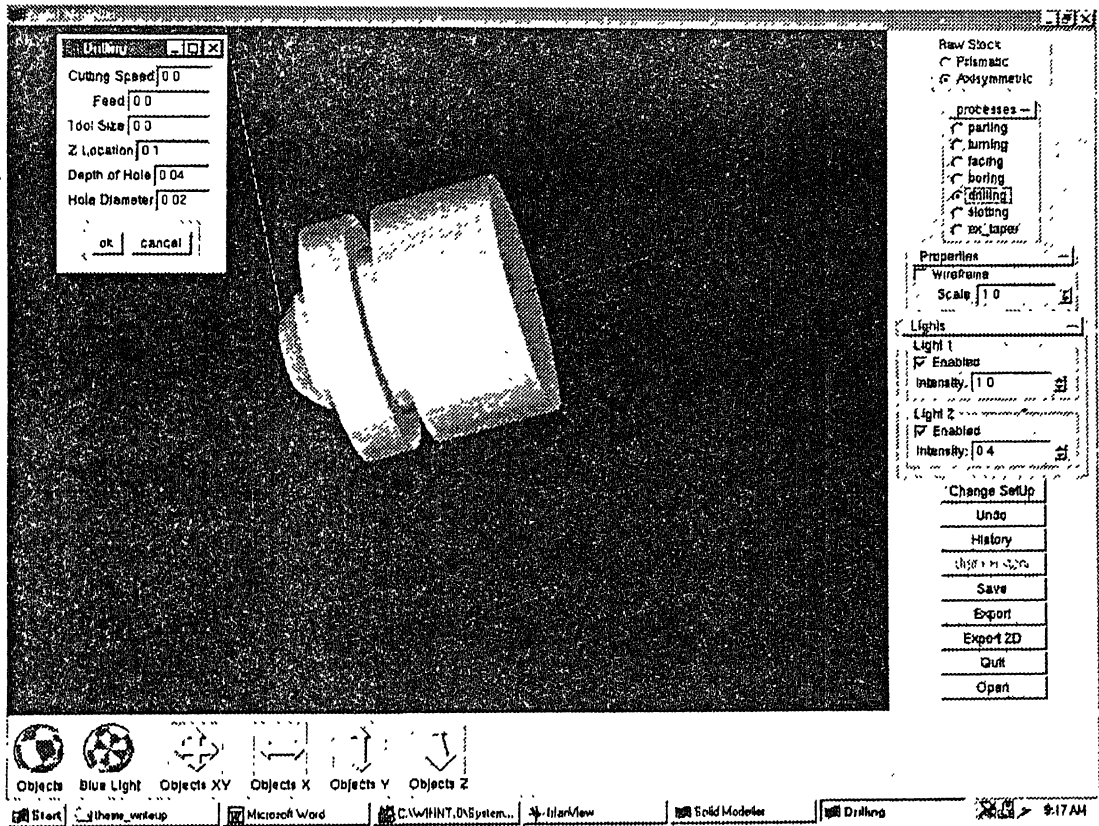
Fig. 4.6 : Change Set Up and Specifying Drilling Parameters

Process Parameters

Cutting Speed : 0.001

Feed   : 0.001

Tool Size   : 0.012

Z Location   : 0.1

Depth of Hole : 0.04

Hole Diameter: 0.02

A Hole is defined by the Depth of Hole and Hole Diameter and its location is given by the Z location.

- **After Drilling and displaying History List**
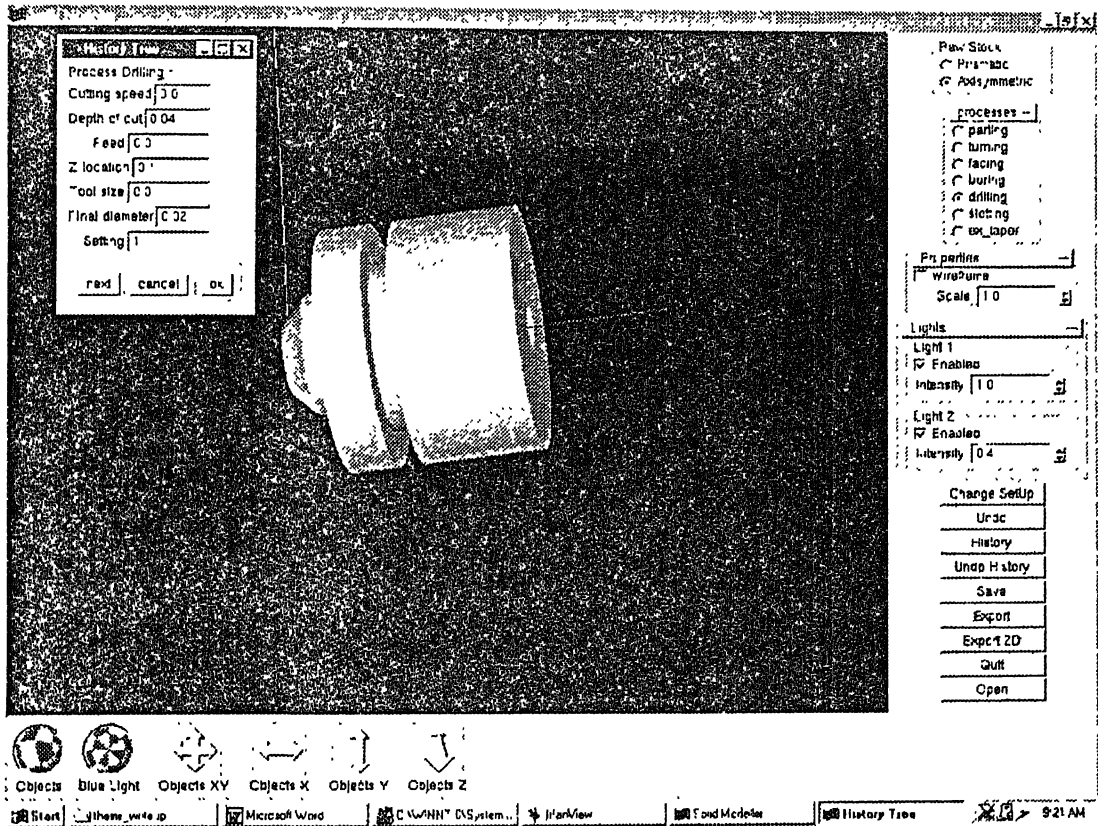
Fig. 4.7a : After Drilling
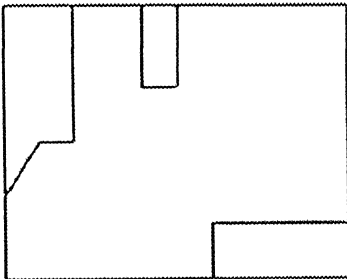
Fig. 4.7b : Final Model Profile

- **After History List Editing**
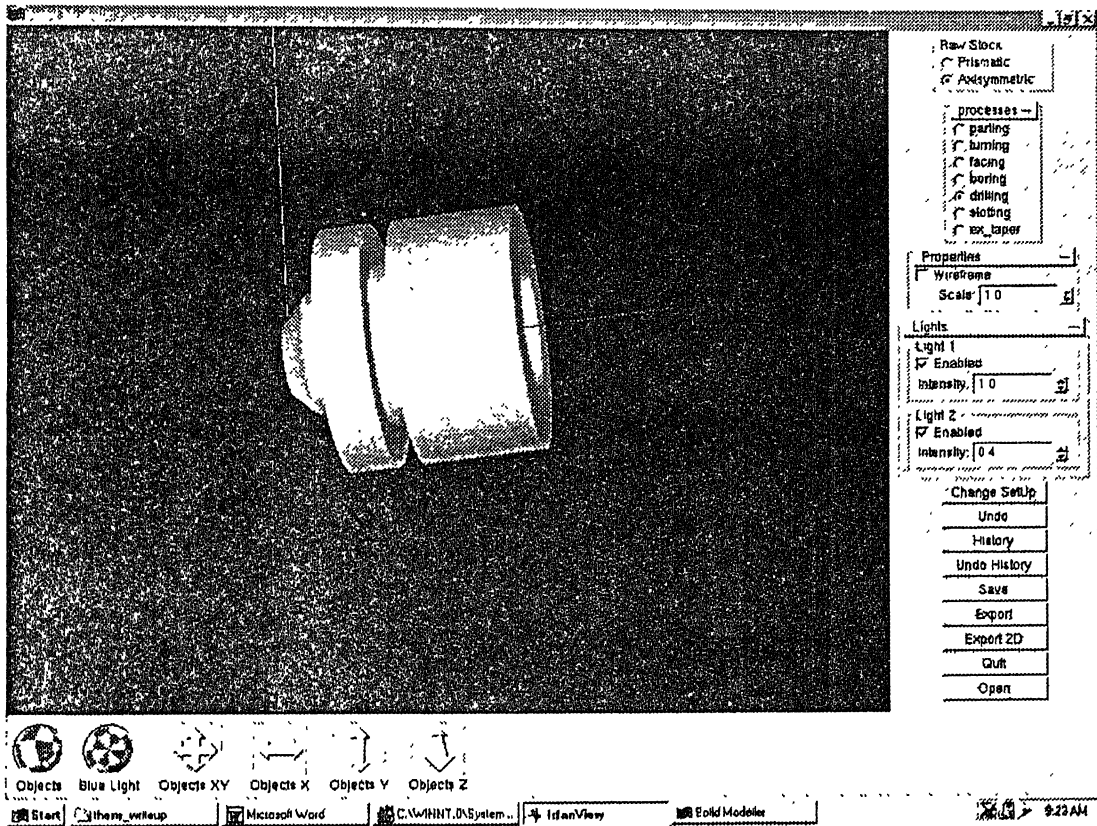
Fig. 4.8 : After History List Editing

In the present case study the hole diameter is modified from 0.02 to 0.05. This change is reflected in the figure 4.8.

Similarly one can modify any previous process parameters when required without need of modelling from the scratch.

Sometimes History List Editing does not produce the logical model and may give absurd results.

Undo History is there!!!

- **WireFrame Display**
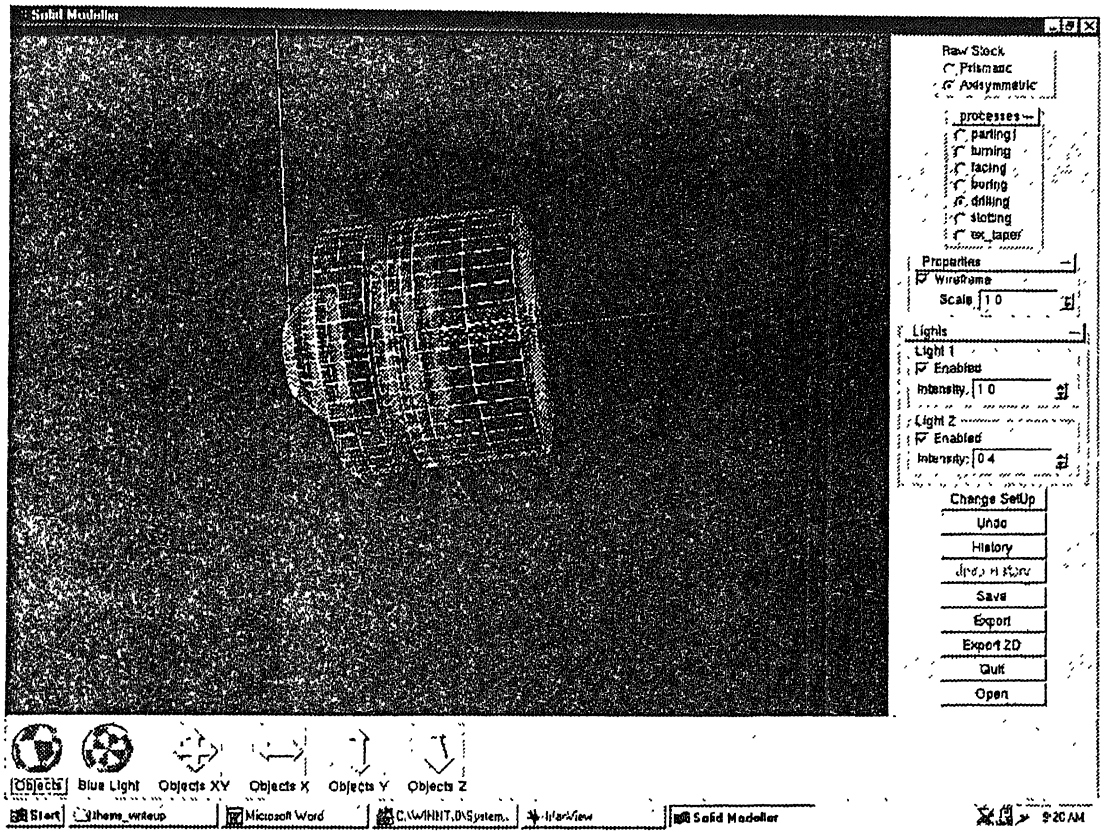
Fig. 4.9 : WireFrame Display

- **Dialog Box For Saving The Data**
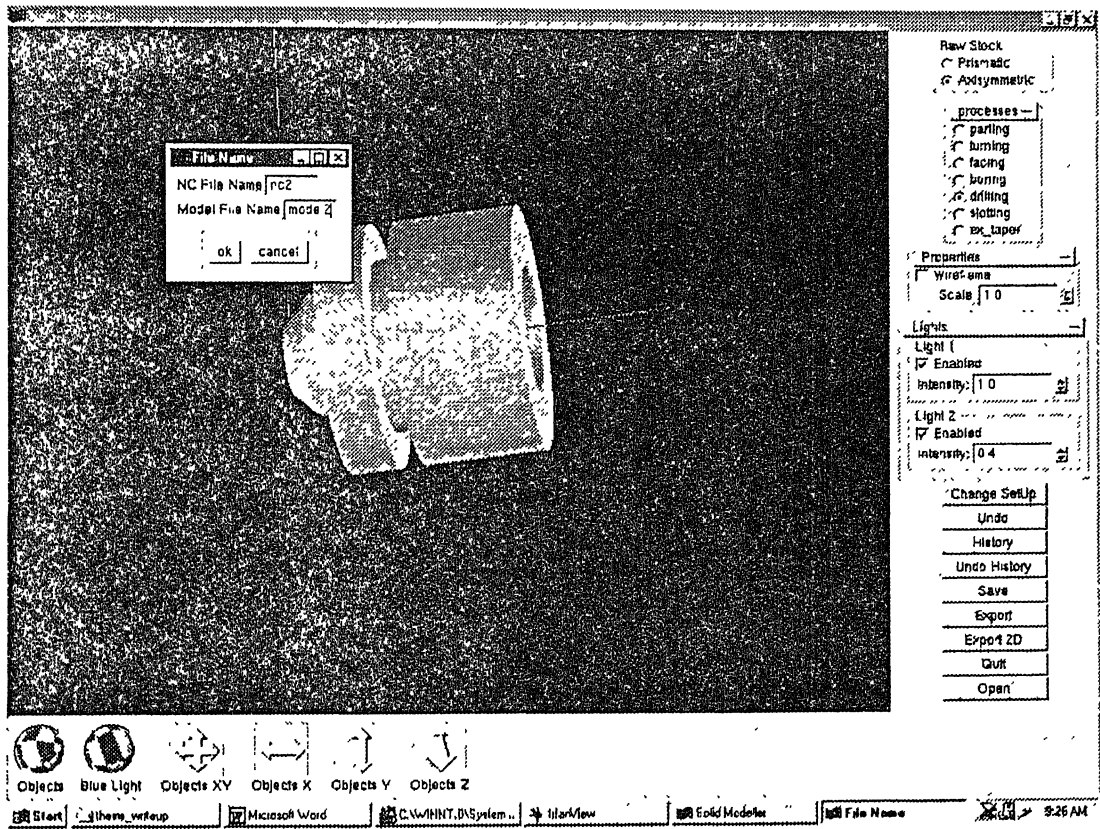
Fig. 4.10 : File Save

Process Information is stored in file "nc2" and Model Information in file "model2".

File "nc2"

Selected stock

Stock material steel
Stock diameter = 0.100000
Stock length = 0.100000

Process 1 Turning :

Process Parameters :

Cutting speed = 0.001
Depth of cut = 0.001
Feed = 0.001
Z Location = 0.100000
Length to cut = 0.020000

Final diameter = 0.050000
Setting = 0

Process 2 Taper Turning :

Process Parameters :

Cutting speed = 0.001
Depth of cut = 0.001
Feed = 0.001
Z Location = 0.100000
Length to cut = 0.010000
Small Diameter = 0.030000
Setting = 0

Process 3 Slotting :

Process Parameters :

Cutting speed = 0.001
Depth of cut = 0.001
Feed = 0.001
Length of parting = 0.01
Z Location = 0.060000
Final diameter = 0.070000
Setting = 0

Process 4 Drilling :

Process Parameters :

Cutting speed = 0.001
Depth of hole = 0.04
Feed = 0.001
Z Location = 0.100000
Tool Size = 0.012
Final diameter = 0.050000
Setting = 1

File "model2"

4
steel
0.100000
0.100000
2
0.00
0.00
0.00
0.100000

- **Dialog Box For Export**

Fig. 4.11 : File Export

File "export2"

```
100
0.000000      0.000000
0.100000      0.000000
0
0.000000      0.000000
0.100000      0.000000
0.100000      0.050000
0.000000      0.050000
10
0.100000      0.050000
0.100000      0.025000
0.080000      0.025000
0.080000      0.050000
0
10
0.100000      0.025000
0.090000      0.025000
0.100000      0.015000
0
10
```

0.060000	0.035000
0.050000	0.035000
0.050000	0.050000
0.060000	0.050000
0	
10	
0.100000	0.000000
0.060000	0.000000
0.060000	0.025000
0.100000	0.025000
1	
1000	

- **Dialog Box For Export 2D**
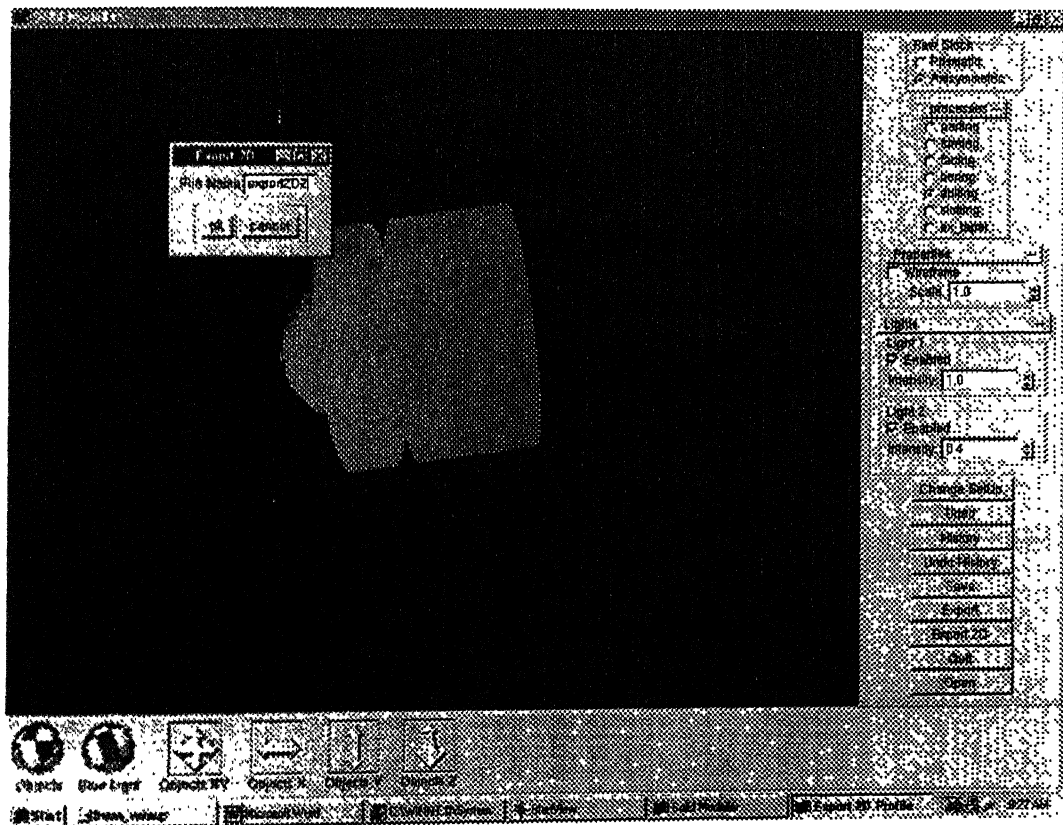
Fig. 4.12 : Export 2D

File "export2D"

0.1000
0.1000
24
0.0000 0.0150

```
0.0100 0.0250
0.0100 0.0250
0.0200 0.0250
0.0200 0.0500
0.0400 0.0500
0.0400 0.0350
0.0500 0.0350
0.0500 0.0500
0.0600 0.0500
0.0600 0.0500
0.1000 0.0500
0.1000 0.0250
0.0600 0.0250
0.0600 0.0000
0.0500 0.0000
0.0500 0.0000
0.0400 0.0000
0.0400 0.0000
0.0200 0.0000
0.0200 0.0000
0.0100 0.0000
0.0100 0.0000
0.0000 0.0000
```

Running profile_generator.c will show the 2D profile of the model.
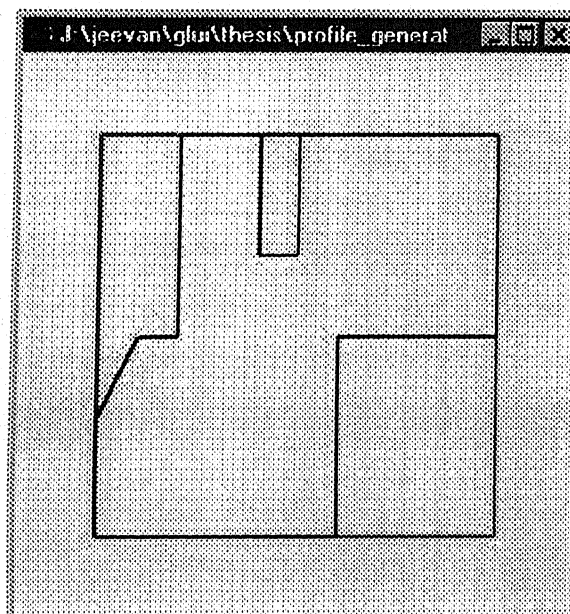
Fig. 4.13 : 2D Profile

- **Sequencing**

It asks for the output file and does sequencing of the machining operations.

Slotting limit is set at 0.005

"sequence" file is as follows :

process is turning
start point = 0.050000 0.035000
end point = 0.040000 0.035000

process = drilling
starting location = 0.100000
Depth of Hole = 0.040000
Diameter of Hole = 0.050000

process is turning
start point = 0.020000 0.025000
end point = 0.000000 0.025000

process = taper turning
start_point = 0.000000   0.015000
end_point = 0.010000   0.025000

- **Some Other Models Created With The Software**

Fig. 4.14 : Example 1

2D Profile

Fig. 4.15 : 2D Profile for Example1

Fig. 4.16 : Example2

2D Profile
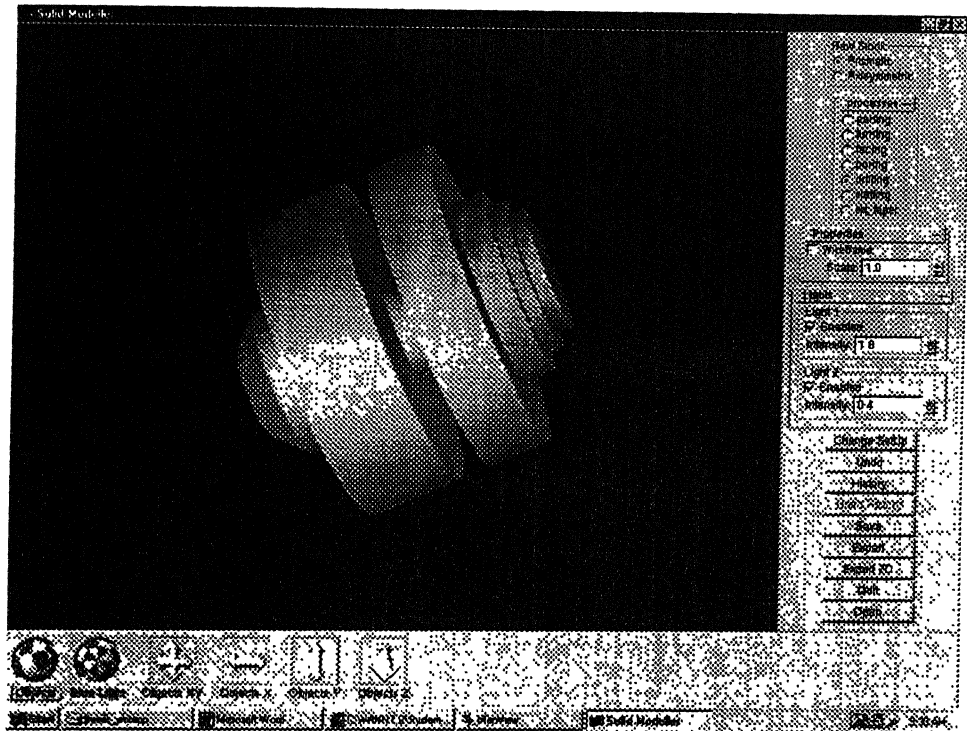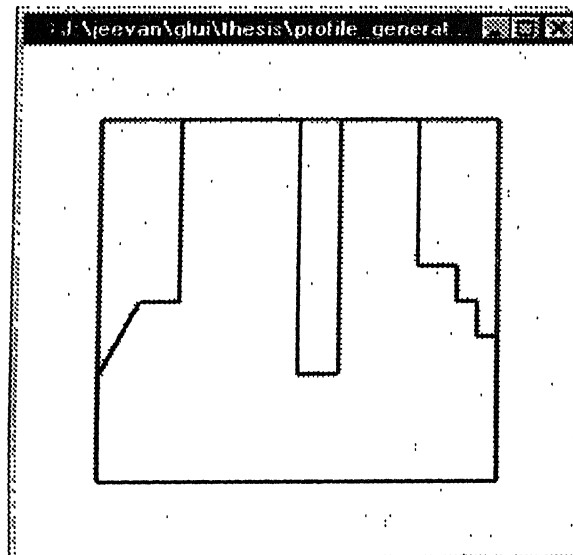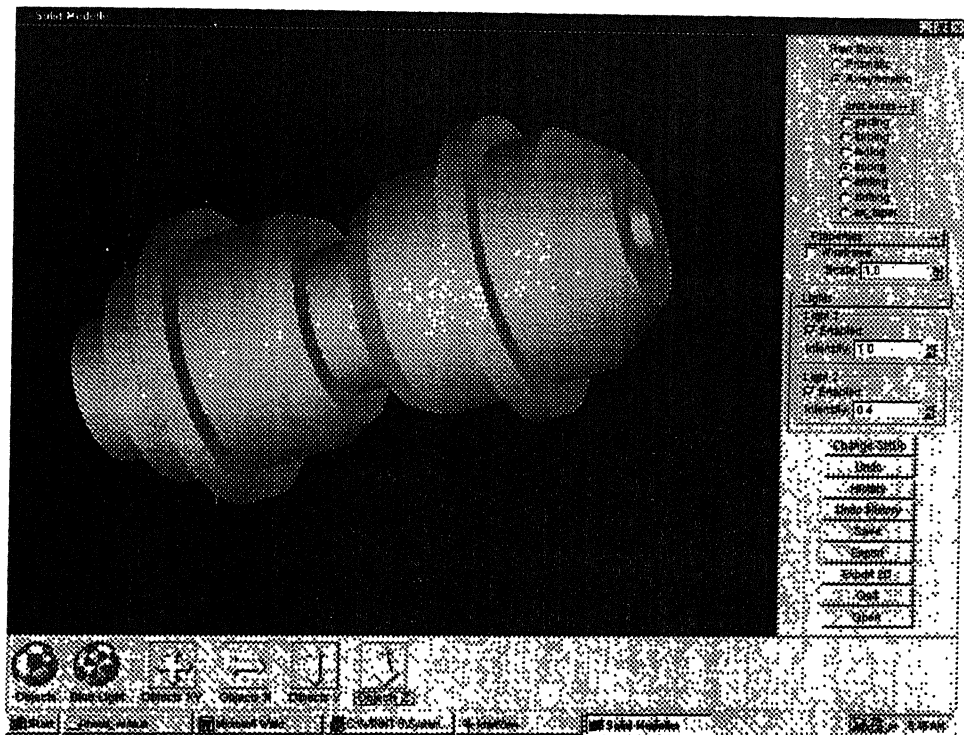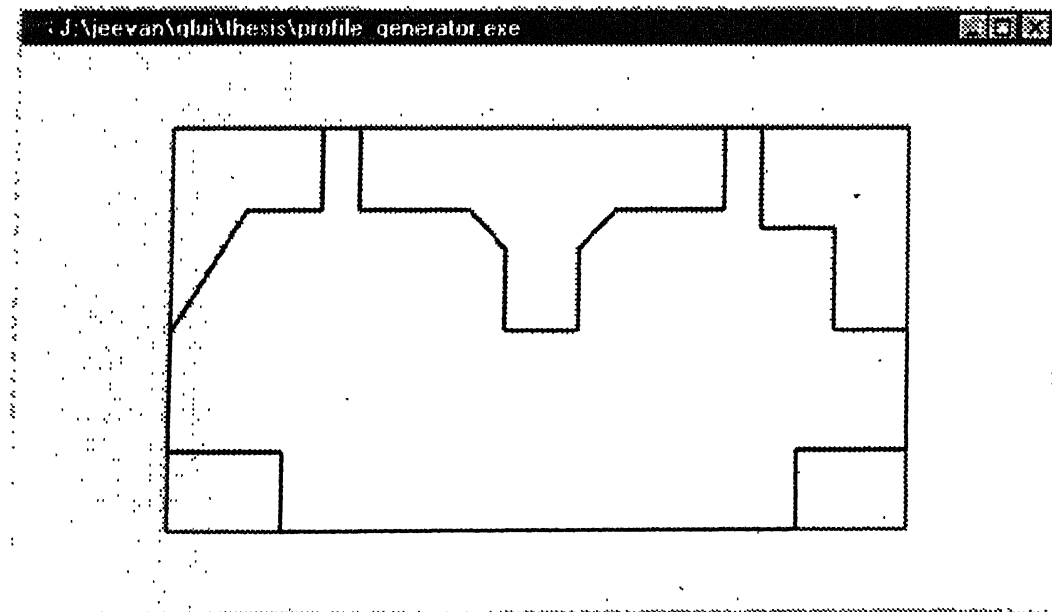

Fig. 4.17 : 2D Profile for Example2

Chapter 5

Conclusion And Scope For Future Work

5.1 Conclusion

The journey of computer aided process planning started with the process of feature recognition going through the stages of different feature recognition algorithms, feature mapping techniques and automated process planning systems. Some tried to develop feature based modelling systems, but every system has it's own advantages and disadvantages. In the present work an attempt has been made to go further towards the process based modeling. With process based modeling, the modeling task becomes simpler and faster. Users need not understand the complex terminology of features and solid modelling.

A Software implementation of process based modeling for axisymmentric parts has been accomplished. Typical lathe processes like turning, facing, parting, drilling, boring, slotting and taper turning have been considered for the modeling. The software provides an easy-to-use graphics user interface and viewing keeping in mind the end user. The terminologies used are simple to comprehend even for laymen. The software provides features like undoing the processes and history tree editing, data storage and retrieval functions. An attempt towards process sequencing has been done once the modellng task is complete so as to make the software compatible to downstream applications like process planning and NC machining. This implementation of process-based modeling is a step towards the goal of having a better integration of various product development activities.

5.2 Scope For Future Work

The present work has been made for axisymmetric parts considering typical lathe based machining operations. The following future work can be taken up to provide for additional modules for the present implementation.

- AI based process sequencing algorithm
- Automated process planning and NC programming
- Data transfer in universal data formats
- Tolerances and other manufacturing attributes should be considered.

These modules will make this software capable of providing a complete platform for modeling, process planning, and interfacing to NC manufacturing for axisymmetric parts. Work towards developing similar platforms for generic part geometries can be taken up.

Bibliography

Ansaldi S, "An edge-face relational scheme for boundary representation", Computer Graphics Forum Vol. 4 (1985) pp 319-332.

Arbab F, "Requirements and architecture of CAM oriented CAD systems for design and manufacture of mechanical parts", Ph. D. Dissertation University of California, 1982.

Ashok Srikantappa and Richard Crawford, "Automatic part coding based on Interfeature Relationships", Manufacturing Research and Technology Vol. 20 (1994) pp 215-237.

Choi B K, Barash M and Anderson D C, "Automatic recognition of machined surfaces from a 3D solid model", Computer Aided Design Vol. 16 No. 2 (1984) pp 81-86.

Chuan-Jun Su, et. al., "An integrated form-feature-based design system for manufacturing", Journal of Intelligent Manufacturing Vol. 6 (1995) pp 277-299.

Cutkosky M Tenenbaum J and Miller D, "Features in process based design", Proc. ASME Computers in Engineering Conf. (1988) pp 557-562.

DeFloriani L, "Feature extraction from boundary models of 3D objects", IEEE Pattern Analysis and Machine Intelligence Vol. 11 No. 8 (1989) pp 785-798.

Henderson M R and Anderson D, "Computer Recognition and extraction of form features", Computers in Industry Vol. 5 (1984) pp 329-339.

Hiroshi Sakurai and Chia-Wei Chin, "Definition and Recognition of volume features for process planning", Manufacturing Research and Technology Vol. 20 (1994) pp 65-81.

Hwang J, " Rule-Based feature recognition: concepts, primitives and implementation", M. S., Arizona State University (1988).

J J Shah and M T Rogers, "Expert form feature modelling shell", Computer Aided Design Vol. 20 No. 9 (1988) pp 515-524.

J J Shah, "Assessment of features technology", Computer Aided Design, Vol. 23 No. 5 (1991) pp 331-343.

Jami J Shah, Yan Shen and Arvind Shirur, "Determination of machining volumes from extensible sets of design features", Manufacturing Research and Technology Vol. 20 (1994) pp 129-159.

Joshi S and Chang T C, "Graph based heuristics for recognition of machined features from 3D solid model", Computer Aided Design, Vol. 20 No. 2 (1988) pp 58-66.

K Tang and T Woo, "Algorithmic Aspects of Alternating sum of volumes. Part 1: Data structure and difference operation", Computer Aided Design, Vol. 23 No. 5 (1991) pp 357-367.

K Tang and T Woo, "Algorithmic Aspects of Alternating sum of volumes. Part 2: Non-convergence and its Remedy", Computer Aided Design, Vol. 23 No. 6 (1991) pp 435-442.

Keith Case, "Using a design by features CAD system for process capability modelling" Computer Aided Design, Vol. 7 No. 1 (1994) pp 39-48.

Kim Y S, "Recognition of form features using convex decomposition", Computer Aided Design, Vol. 29 No. 9 (1992), pp 461-476.

Kim Y. S. and Wilde D. J, " A convergent Convex Decomposition of Polyhedral objects" ASME Journal of Mechanical Engineering Vol. 114 (1992) pp 468-476.

Kyprianou L "Shape classification in Computer Aided Design", Ph.D. Dissertation, university of Cambridge (1980).

Murray J.L. and Yue Y, " Automatic machining of 2.5D components with the ACIS modeller", International Journal of Computer Integrated Manufacturing, Vol. 6 (1993), pp 94-104.

Peter T. J., "Encoding mechanical Design features for recognition via neural nets", Technical Report No. CSE-TR-91-20). Department of Computer Science, University of Connecticut (1991).

S Prabhakar and M R Henderson, "Automatic from feature recognition using neural network based techniques on boundary representations of solid models", Computer Aided Design, Vol. 24 No. 7 (1992) pp 381-393.

Sakurai S and Gossard D, "Shape feature recognition from 3D solid models", Proc. ASME computers in engineering Conf. (1988) pp 515-519.

Timo Laakko and Martti Mantyla, "Feature modelling by incremental feature recognition", Computer Aided Design, Vol. 25 No. 8 (1993) pp 479-492.

Turner G P and Anderson D C, "An object oriented approach to interactive feature based design for quick turn around manufacturing", Computational Engineering Vol. 1 (1988) pp 551-555.

V Sundararajan and Paul K Wright, "Identification of multiple feature representations by volume decomposition for 2.5 D components", Transactions of the ASME Vol. 122 (2000) pp. 280-290.

Vanderbrande and Requicha, "Geometric computation for the recognition of spatially interacting machining features", Manufacturing Research and Technology (1994) pp 83-107.

W Duan , J Zhou and K Lai, "FSMT : a feature solid-modelling tool for feature-based design and manufacture", Computer Aided Design, Vol. 25 No. 1 (1993) pp 29-37.